

Volume 1
Issue 9

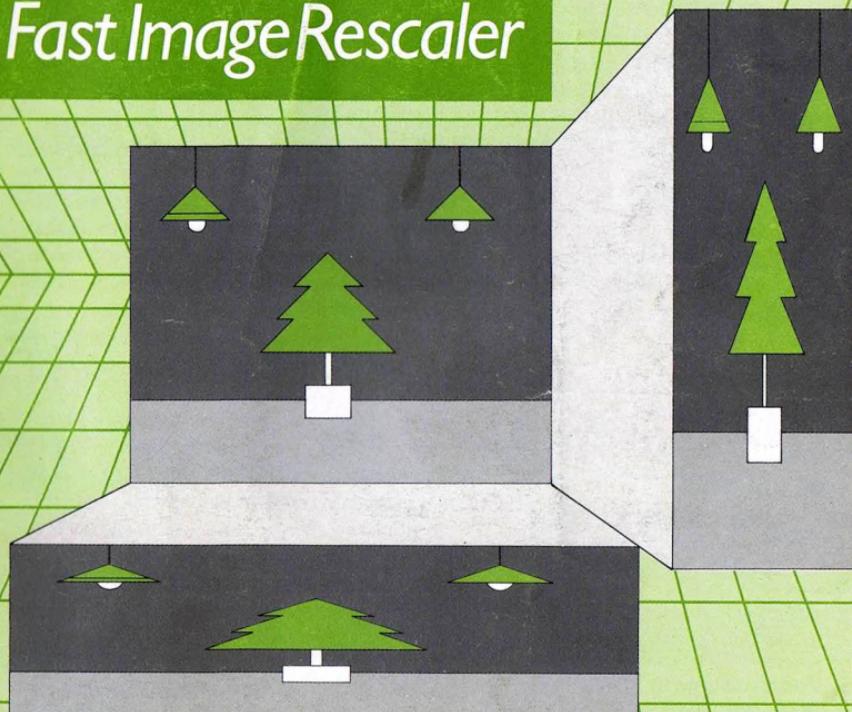
September
1988

Price £1.20

RISC USER



A Fast Image Rescaler



THE MAGAZINE AND SUPPORT GROUP
EXCLUSIVELY FOR USERS OF THE ARCHIMEDES

CONTENTS

FEATURES

News	4
Animating Archie (Part 2)	11
Basic V - The Error of its Ways (Part 2)	14
It's more than Hearsay	21
Using the PC Emulator (Part 2)	22
Archimedes Visuals	24
Introducing ARM Assembler (Part 5)	31
Postbag	44
Hints & Tips	45

UTILITIES AND APPLICATIONS

3D Landscape Editor	5
Music Maestro	7
A Fast Image Rescaler	15
An ADFS File-Find Utility	27
RISC User Toolbox (Part 3)	35
Fast Module Utility	39
Screen Freezer and Dumper (Update)	42

REVIEWS

Hold the Front Page	18
Plotting with Minerva	28
Arcterm Professional	40

RISC User is published by BEEBUG Ltd.
Co-Editors: Mike Williams, Dr Lee Calcraft
Assistant Editor: Kristina Lucas
Technical Editor: David Spencer
Production Assistant: Yolanda Turuelo
Technical Assistant: Lance Allison
Subscriptions: Mandy Mileham

BEEBUG, Dolphin Place, Holywell Hill, St.Albans, Herts
AL1 1EX. Tel. St.Albans (0727) 40303

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, BEEBUG Limited.

BEEBUG Ltd (c) 1988



The Archimedes Magazine and Support Group.

EDITORIAL

As we approach the first anniversary of the launch of RISC User we are very pleased by the response we have received to the magazine. By the time you have received this issue we anticipate that RISC User membership will have grown to be significantly in excess of 5000. At the same time the magazine has expanded from its original 32 pages to its present 48. All of this reflects a steadily growing interest in the Archimedes by the computer world at large.

For example, the Newsprint columns of Personal Computer World, which have often been critical of Acorn in the past, have more recently portrayed the Archimedes in a much more favourable light. Further, a meeting of Mandelbrot enthusiasts took place in July this year at the Delft University of Technology in the Netherlands, where the Mandelbrot program by Stephen Streeter (featured on the magazine disc for Issue 8) impressed everyone present by its speed and the speed of the Archimedes in general. These are just two examples of the growing reputation which the Archimedes is earning for itself.

For the future, we are currently working on a disc to celebrate the start of volume 2 of RISC User. The details have still to be finalised but it is intended that this will contain some of the best programs from RISC User (updated where appropriate), outstanding visual demonstrations and a complete computerised index to the whole of volume 1 - all designed to show just what the Archimedes is capable of. This disc will be available at a special low price to RISC User members. We shall also be producing a complete printed index to volume 1 which will be mailed out to all members with the first issue of volume 2. All in all we believe that we can continue to offer even more extensive and varied support for Archimedes users in the future.

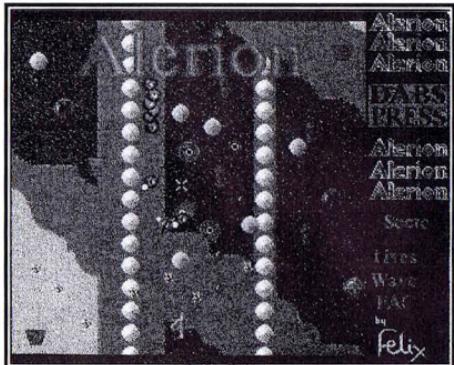
RISC User Reviews

In the interests of fairness to all, we do not publish comparative reviews of our own products. With BEEBUG's recently released comms package for the Archimedes, called *Hearsay*, we have instead invited David Pilling, author of the software, to write his own account of what the package has to offer, rather than compare this directly with *ARCterm* or other comms software.

News News News News News

ALERION INVASION

Alerion is the first game for the Archimedes from Dabs Press, a company better known for its computing books and utilities software. *Alerion* (it means an eagle displayed without feet or a beak) follows the style of many other 'shoot-em-up' games, where the object is to fly your craft over a landscape of 'nasties' which try to kill you. You survive by dodging the enemy, or destroying them with your bombs and machine gun. *Alerion*, which is supplied on a copy-protected 800K disc, costs £14.95 inc. VAT and is available directly from Dabs Press, 76 Gardner Road, Prestwich, Manchester M25 7HU, phone 061-773 2413, or from BEEBUG Retail.



PC LASER CARD USES ARM

In the first of a number of planned specialist applications of the ARM chipset, Acorn has released a card for driving laser printers. The new board, which is primarily aimed at other manufacturers, allows any IBM PC or compatible, to be connected to a dumb laser printer engine. Depending on the software that is used, the laser printer can be made to emulate any particular printer, or to support a full Page Description Language such as *PostScript*. Until now, most laser printers have been slowed down by the speed at which the controller can operate, but the speed of the ARM chip allows the printer engine to run at full speed, printing 8-10 pages a minute. Hopefully, this will be just the first of many specialist applications for the ARM, as widespread use of the chipset can only be beneficial to Archimedes owners.

EFFECTS GALORE

Archeffect is a visual effects system for the Archimedes from a new company called 'FX'. *Archeffect* is supplied on disc as a relocatable module, together with a demonstration program. The package allows mode 12, 13, 14 and 15 screens to be manipulated in a variety of ways. For example, screen pictures can be rolled down the screen, rolled off one edge, turned upside down, reduced in size, or distorted into a wine-glass shape. All the various effects are controlled by simple star commands, and can be used very easily from Basic to produce a host of effects. *Archeffect* comes with a 42 page manual and costs £24.99 inclusive. It is available from FX, 207 South Avenue, Southend-on-Sea, Essex SS2 4HT, or phone (0702) 68748.

BUFFER YOUR DATA

The latest product from Clares Micro Supplies is an Archimedes module that extends the printer and serial buffers. This means that you can use the computer for other things while printout continues. The size of the additional buffer can be chosen from a few bytes, right up to four megabytes. The contents of the buffer are cleared on a hard reset, but otherwise, data can be entered into the buffer and then forgotten about. So for example, you could print a large document using the buffer, and start writing some new text while the original is still printing. The buffer software costs £19.95 inclusive from Clares Micro Supplies, 98 Middlewich Road, Rudheath, Northwich, Cheshire CW9 7DA, phone (0606) 48511.

KEEP TRACK OF YOUR TEXT

ArcTFS from Texcellence is a new Archimedes database system, designed especially for storing large amounts of text. *ArcTFS* allows text files to be imported from most word processors, or typed in directly, and then indexed in a variety of ways. This allows rapid access to any piece of text, as well as offering such features as fast keyword searching for particular topics. One application for *ArcTFS* is in the storing of data by researchers, who need quick access to all available information on a given subject. *ArcTFS* is supplied on an 800K disc for £29.95, and can be obtained from Texcellence, 2 Greenhill Road, Coleraine, Northern Ireland BT51 3JE.

3D LANDSCAPE EDITOR



Gary Smith's impressively short program enables you to create and edit multi-coloured 3D landscapes.

Screen size must be set to 20 on 300 series machines.

No doubt nearly all Archimedes owners have seen and marvelled at the colourful 3D landscape in Zarch and its successor Conqueror. In fact, producing such a landscape is relatively straightforward, as the accompanying program illustrates, though to make this scroll dynamically is another problem altogether.

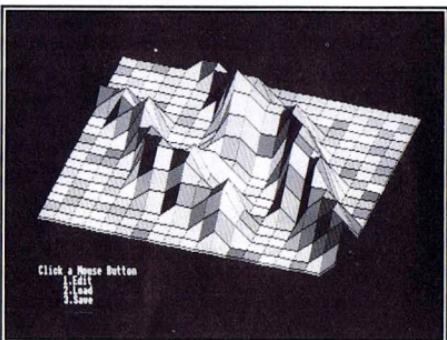
The purpose of this program is to allow you to create, and subsequently edit a 3D landscape laid out on a 20x20 grid. Landscapes created in this way may also be saved and recalled for further modification. Although the program is a complete landscape editor, the ideas it contains and their implementation are also likely to be of interest.

Type the program in and save it away. When you run it, a simple message at the foot of the screen first prompts for the file name of an existing landscape. Just press Return and you will then see displayed the default flat landscape. This provides a useful starting point.

At this point, you have the choice of editing the current display, saving the data, or loading new data, by pressing the left, middle or right-hand buttons on the mouse respectively. If you select edit mode, the multi-coloured display is replaced by a 20x20 grid of numbers. These can be in the range 0 to 9 to indicate height, and may be modified by moving the mouse pointer to the required digit and clicking the left-hand button to increment, or the middle button to decrement that figure. At any time, pressing the right-hand button will return you to the 3D display.

The colours used are specified in the DATA statement at line 130. Colours are then used randomly, but the landscape is illuminated as though any light were coming from the right, and dark and light shades are chosen accordingly. Note too, that the numbers in the grid refer to the points at the intersection of the

grid lines, hence the coloured squares themselves form a 19x19 grid. It is probably well worth while trying a few simple effects until you have mastered the technique.

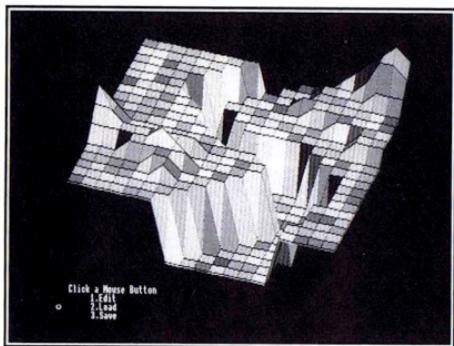


When the 3D version is displayed you may choose to save the current data, or to load new data, as well as to edit the landscape further. Both load and save options prompt for a file name (Escape aborts either). At other times Escape terminates execution of the program as a whole. The data is stored in an array (D%). When a new display is to be created, PROCcalc calculates the appropriate values which are stored in the two arrays X% and Y%. PROCdisplay then uses this information to create the landscape.

A couple of points are worth noting. The grid lines themselves may be omitted from the 3D display by deleting line 970 from the program. Try this and see which version you prefer. Also, if you find that the numbers change too quickly when editing the data, increase the value at the end of line 1140.

A number of complete landscapes have been included on this month's magazine disc. Use these as a guide as to what can be achieved, and as the basis of your own landscapes if you wish. Although the program has a serious purpose, we are also sure that you will have fun creating your own fictional landscape.

3D LANDSCAPE EDITOR



```

10 REM >LandScape
20 REM Program Landscape Editor
30 REM Version A 1.5
40 REM Author Gary Smith
50 REM RISC User September 1988
60 REM Program Subject to copyright
70 :
80 MODE 15:ON ERROR PROCerror:END
90 OFF:@%=
100 DIM D%(20,20):DIM colour(17)
110 DIM X%(20,20),Y%(20,20)
120 FOR F%=1 TO 17:READ colour(F%):NEXT
T
130 DATA 4,5,9,8,24,25,12,28,13,30,29,
31,14,44,45,46,47
140 PROCload
150 REPEAT
160 IF new% THEN PROCCalc:PROCdisplay
170 REPEAT:MOUSE x%,y%,b%:UNTIL b%=0
180 PRINT TAB(0,28); "Click a Mouse But
ton"
190 PRINTTAB(5)"1.Edit"
200 PRINTTAB(5)"2.Load"
210 PRINTTAB(5)"3.Save";
220 MOUSE x%,y%,b%
230 CASE b% OF
240 WHEN 4:PROCedit
250 WHEN 2:PROCload
260 WHEN 1:PROCsave
270 ENDCASE
280 UNTIL FALSE
290 END
300 :
310 DEF PROCerror
320 MODE12:REPORT:PRINT" at line ";ERL
330 ENDPROC
340 :
350 DEFPROCsave
360 LOCAL ERROR
370 VDU28,0,31,30,28,12:
380 ON
390 ON ERROR LOCAL GOTO460
400 PRINTTAB(0,3); "Enter File Name to
Save"
410 INPUT">>"name$"
420 Z% =OPENOUT(name$)
430 FOR F%=1 TO 20:FOR G%=1 TO 20
440 BPUT#Z%,D%(F%,G%)
450 NEXT:NEXT
460 CLOSE#0:new% =FALSE:VDU12,26:OFF
470 ENDPROC
480 :
490 DEF PROCload
500 LOCAL ERROR
510 VDU28,0,31,30,28,12:new% =FALSE:ON
520 ON ERROR LOCAL IF ERR=17 THEN GOTO
650
530 PRINTTAB(0,3)"Enter File Name to L
oad"
540 INPUT">>"name$"
550 new% =TRUE
560 IF name$ <> "" THEN
570 Z% =OPENUP(name$)
580 IF Z% <> 0 THEN
590 FOR F%=1 TO 20:FOR G%=1 TO 20
600 D%(F%,G%) =BGET#Z%
610 NEXT:NEXT
620 ELSE PRINT"No such file":new% =FALS
E:Z% =INKEY(100)
630 ENDIF
640 ENDIF
650 CLOSE#0:VDU12,26:OFF
660 ENDPROC
670 :
680 DEF PROCCalc
690 FOR F%=0 TO 19:FOR G%=0 TO 19
700 A% =D%(F%+1,G%+1):x% =F%*50:y% = (21-F
%)*10
710 x% +=G%*15:y% +=G%*25
720 y% +=A%*40
730 X%(F%+1,G%+1) =x%:Y%(F%+1,G%+1) =y%
740 NEXT:NEXT:ENDPROC
750 :
760 DEF PROCprint_values
770 CLS
780 FOR F% =1 TO 20:PRINTTAB(F%*3)F%:;NEX
T
790 FOR F% =1 TO 20:PRINT'F%:;FOR G% =1 TO 2
0

```

Continued on page 42



MUSIC MAESTRO

by Crosbie Fitch

This short program by the author of the Welcome Music Editor will replay any piece of music created with the editor, and can be customised to your own requirements.

Although the Music Editor on the Welcome disc is a sophisticated and powerful piece of software, there are occasions when it may seem too much of a good thing if, for example, you just want to play a piece of music from a file. Again, there is no way in which a repertoire of pieces to be played can be set up in advance. All that is possible with the Music Player listed here, which will play back any piece of music created and saved using the Music Editor.

Type the program in and save it away. When run it prompts for a music file, loads it from disc and then plays it. It is as simple as that. Although some of the code is quite difficult to understand, it is only lines 70 to 180 which you need to follow. These can be readily adapted for other needs.

The program as listed includes a loop so that when one piece of music finishes playing, the program prompts for the name of the next piece. You could also modify the program to set up a musical carousel which would automatically play through a whole musical programme set up in advance. In fact, the music uses up such little time within the main loop that there is plenty of opportunity to include other tasks as well and have the music playing in the background. This is an area which will surely repay any experimenting.

To help you get started we have included a Mozart Allegro on this month's disc. Don't forget too, that a set of miniature stereo headphones plugged into the back of your Archimedes will improve the sound quality considerably, and increase the volume!

Happy listening.

```
10 REM >AnyPlay
20 REM Version A 1.2
30 REM Author Crosbie Fitch
40 REM RISC User Sept 1988
50 REM Program subject to copyright
60 :
70 PROCinitialise
80 REPEAT
90 INPUT"Enter tune: " Tune$
100 PROCload_music(Tune$)
110 PROCplay_start
120 REPEAT
130 IF Playing% THEN
140 B1%=B2%:B2%=BEAT
150 IF B2%<B1% PROCplay_bar
160 ENDIF
170 UNTIL NOT Playing%
180 UNTIL FALSE
190 END
200 :
210 DEF PROCload_music(F$)
220 LOCAL T%,F%,C%,A%
230 SYS"OS File",5,F$ TO F%,,,C%,A%
240 IF (F%-1)ANDA% AND 1 AND C%>8 THEN
250 F%=OPENIN(F$):F$=""
260 FOR C%=1 TO 7:F$+=CHR$BGET#F%:NEXT
270 C%=BGET#F%
280 IF F$="Maestro" THEN
290 CASE BGET#F% OF
300 WHEN 0
310 WHEN 1:PROCtempo:PROCLinstruments
320 PROCLstaves:PROCLmusic:T%:TRUE
330 OTHERWISE A%:FALSE
340 REPEAT ON BGET#F% PROCLmusic,PROCL
staves,PROCLinstruments,PROCLvolumes,PROCL
stereos,PROCtempo ELSE A%:TRUE
350 UNTIL EOF#F% OR A%:T%:TRUE
360 ENDCASE
370 ENDIF
380 CLOSE#F%
390 ENDIF
400 IF T% ELSE ERROR214,F$+" file coul
d not be opened"
410 ENDPROC
```

TECHNICAL NOTE: As its name implies, the procedure PROCinitialise at line 70 should only ever be executed once at the start of the program.



MUSIC MAESTRO

```

420 :
430 DEF PROCIMusic
440 LOCAL B%,S%:INPUT#F%,S%
450 DIM Music% S%:Gate%=Music%+S%
460 FOR C%=0 TO 7:INPUT#F%,S%
470 DIM B% S%:Music%(C%)=B%
480 Fine%(C%)=B%+S%
490 NEXT
500 B%=Music%
510 WHILE B%<Gate%
520 ?B%=>BGET#F%:B%+=1
530 ENDWHILE
540 FOR C%=0 TO 7:B%=Music%(C%)
550 WHILE B%<Fine%(C%)
560 ?B%=>BGET#F%:B%+=1
570 ENDWHILE
580 NEXT
590 ENDPROC
600 :
610 DEF PROCStaves
620 Stave%=>BGET#F%:PERC%=>BGET#F%
630 FOR C%=0 TO 7
640 S_C%(C%)=Stave_Channels%(Stave%,C%
)
650 NEXT:C%=0
660 WHILE C%<PERC%:C%+=1
670 S_C%((7-(2-C%)*(5-Stave%))=Stave%+C%
%
680 ENDWHILE
690 ENDPROC
700 :
710 DEF PROCLInstruments
720 FOR C%=0 TO 7
730 SYS"Sound_AttachVoice",BGET#F%+1,((
BGET#F%-1)MODNVoices%+1
740 NEXT
750 ENDPROC
760 :
770 DEF PROCLVolumes
780 FOR C%=0TO7:Volumes%(C%)=BGET#F%:N
EXT
790 ENDPROC
800 :
810 DEF PROCLStereos
820 FOR C%=0 TO 7
830 STEREO C%+1,Stereo%(BGET#F%)
840 NEXT
850 ENDPROC
860 :
870 DEF PROCLTempo:Tempo%=BGET#F%
880 TEMPO Tempo%(Tempo%)*128*4096DIV60
0
890 ENDPROC
900 :
920 DEF PROCplay_start
930 LOCAL C%:PBAR%=1
940 PP%=>Music%+2:P%()=Music%()
950 FOR C%=0 TO 3:PCLEF%(C%)=Clef%(0):
NEXT
960 PROCplay_key_sig(2):Playing%=>TRUE
970 SYS"Sound_QInit"
980 Beats%=>4*Length%(12):Q%()=(Beats%)
990 TIE%=>FF:B2%=&10000
1000 BEATS Beats%
1010 SYS"Sound_QSchedule",Beats%,&F0401
C5,Tempo%(Tempo%)*128*4096DIV6000
1020 C%=>Beats%/50*&1000
1030 IF C%>&7FFF C%=&7FFF
1040 TEMPO C%
1050 ENDPROC
1060 :
1070 DEF PROCplay_bar
1080 LOCAL C%,L%,I%,D%,S%,Q%,T%,B%,A%
1090 Q%()=(Beats%):B%=>PBAR%
1100 Accidental%()=(0)
1110 WHILE B%=>PBAR% AND PP%<Gate%
1120 IF PP%>PROCplay_notes(PP%):PP%+=
1 ELSE PROCplay_attribute(PP%?1):PP%+=2
1130 ENDWHILE
1140 IF PP%>=Gate% Playing%=>FALSE
1150 ENDPROC
1160 :
1170 DEF PROCplay_notes(G%)
1180 Q%=>FALSE:C%=>TRUE
1190 REPEAT
1200 REPEAT:C%=>TRUE:UNTIL G%AND%1<<C%
1210 IF Q%(S_C%(C%))>Q% Q%=>Q%(S_C%(C%))
1220 UNTIL (2<<C%)>G%
1230 Q%()=(\$10000):C%=>TRUE
1240 REPEAT
1250 REPEAT C%=>TRUE:UNTIL G%AND%1<<C%
1260 T%=>P%(C%):D%=>P%(C%):?1:I%=>D%>>3
1270 S%=>S_C%(C%):L%=>T%>>3:A%=>0
1280 IF L% AND S%<=Stave% THEN
1290 IFD%AND7 Accidental%(S%,L%)=D%AND7
1300 A%=>Accidental%(S%,L%)
1310 L%+=>PCLEF%(S%)
1320 IF A% ELSE A%=>Key%(L%MOD7)
1330 ENDIF
1340 IF TIE% AND %1<<C% THEN

```



```

1350 D%Duration%(Tempo%)?I%
1360 IF T% AND 4 TIE%==TIE% AND NOT(%1<<
C%):T%==P%(C%)+1:REPEAT%+=2:D%+=Duration
%(Tempo%)?(??%>>3):UNTIL T%>Fine%(C%) OR
4 AND NOT T%?TRUE:IF D%>254 D%==254
1370 IF L% SOUND C%+1,Volume%(Volumes%(C%))
OR&100,Line(L%)+Aoff(A%),D%,Q%
1380 ELSE
1390 IF 4 AND NOT T% TIE%==TIE% OR%1<<C%
1400 ENDIF
1410 P%(C%)+=2
1420 IF Length%(I%)<QI%(S%) QI%(S%)=Len
gth%(I%):Q%(S%)=Q%+QI%(S%)
1430 UNTIL (2<<C%)>G%
1440 ENDPROC
1450 :
1460 DEF PROCplay_attribute(A%)
1470 C%==TRUE
1480 REPEAT C%==TRUE:UNTIL A%AND%1<<C%
1490 ON C%+1 PROCplay_time_sig(A%),PROC
play_key_sig(A%),PROCplay_clef(A%),,PROC
play_bar_line(A%)
1500 ENDPROC
1510 :
1520 DEF PROCplay_time_sig(A%)
1530 A%=(A%>>1 AND &F)+1)*Length%(A%>>
3 AND %11100)
1540 SYS"Sound_QSchedule",Beats%,&F0401
C6,A%:Beats%==A%
1550 ENDPROC
1560 :
1570 DEF PROCplay_key_sig(A%)
1580 LOCAL N%:A%==A%>>2
1590 FOR N%==0 TO 6:Key%(N%)=Key_Sig%(A%
,N%):NEXT
1600 ENDPROC
1610 :
1620 DEF PROCplay_clef(A%)
1630 PCLEF%(A%>>6)=Clef%(A%>>3 AND 3)
1640 ENDPROC
1650 :
1660 DEF PROCplay_bar_line(A%)
1670 PBAR%+=1
1680 ENDPROC
1690 :
1700 DEF PROCinitialise
1710 LOCAL N%,C%,D%,ST
1720 SOUND ON:VOICES 8
1730 DIM Stereo%(6),Tempo%(14),Line(42)
,Aoff(7),Clef%(3),Key%(6),Key_Sig%(15,6)
1740 DIM Length%(31),Duration%(14),Acci
dental%(3,31),Q%(5),QI%(5),Music%(7)
1750 DIM Fine%(7),P%(7),PCLEF%(3),Stave
_Channels%(3,7),S_C%(7),Volume%(8)
1760 DIM Volumes%(7),Key_Y%(3,1,6)
1770 FOR N%=-3 TO 3:Stereo%(N%+3)=(2^(5
+ABSN%)-1)*SGNN%:NEXT
1780 FOR N%==0 TO 14:READTempo%(N%):NEXT
1790 DATA 40,50,60,65,70,80,90,100,115
1800 DATA 130,145,160,175,190,210
1810 ST=$1000/12
1820 FOR N%==0 TO 42:Line(N%)=(1+N%DIV7<
<12)+(ASCMID$("024579;",N%MOD7+1)AND15)*
ST+.49:NEXT
1830 Aoff(2)=ST:Aoff(3)=-ST
1840 Aoff(4)=ST*2:Aoff(5)=-ST*2
1850 Aoff(6)=ST:Aoff(7)=-ST
1860 Clef%(0)=11:Clef%(1)=5
1870 Clef%(2)=3:Clef%(3)=-1
1880 FOR C%==TO03:FOR N%==TO1:FORD%==TO6
1890 Key_Y%(C%,1-N%,D%)=3*(D%AND%1)-D%D
IV2+(D%-3)*N%+(N%ANDC%>>2AND(D%AND5)=0)*
7-1-(C%-1>>1)-2*(C%==2)
1900 NEXT:NEXT:NEXT
1910 FOR C%==TO15:FOR N%==TO(C%>>1)-1
1920 Key_Sig%(C%,(7+Key_Y%(1,C%AND%1,N%
))MOD7)=C%MOD2+2
1930 NEXT:NEXT
1940 FOR C%==TO31:Length%(C%)=(%1<<7-(C
%>>2))*(%1111000>>(C%AND3)AND%1111):NEXT
1950 FOR N%==0 TO 14:DIM C% 32
1960 Duration%(N%)=C%
1970 FOR C%==0 TO 31
1980 D%==75/Tempo%(N%)*Length%(C%)/8+.5:
IF D%>254 D%==254
1990 Duration%(N%)?C%D%
2000 NEXT:NEXT
2010 FOR N%==0 TO 3:FOR C%==0 TO 7
2020 Stave_Channels%(N%,C%)=(N%+1)*C%DI
V8
2030 NEXT:NEXT
2040 Stave_Channels%(2,1)=1
2050 Stave_Channels%(2,2)=1
2060 Stave_Channels%(2,5)=2
2070 SYS"Sound_InstallVoice"TO ,NVoice
s%:NVoice%==1
2080 FOR N%==0 TO 8:Volume%(N%)=(N%+1)*1
28/9-1:NEXT:Volumes%()=(8)
2090 FOR C%==1 TO 8:STEREO C%,0:NEXT
2100 ENDPROC

```

DABS PRESS

Dabhand User News

Alerion is the first traditional shoot-em-up game for the Archimedes brought to you by the experts: David Atherton and Bruce Smith. Written in ARM machine code it uses the spectacular speed, sound and colour of the Archimedes. We reckon that it's impossible to finish and will give you hours of addictive fun for just £14.95.

If you're a serious user then our 368 page Dabhand Guide on Archimedes Assembly Language is an absolute must and is packed full of information and advice on programming the Archie. If you are interested in getting more power and more speed from your Archimedes then don't miss our stand at the PC Show (formerly the PCW Show). We'll be previewing a major piece of software for the Archie that will change the way in which you program. As easy as ABC!

Dabs Press, Stand 2132 PC Show, 14th to 18th September 1988 inclusive. Don't miss us!

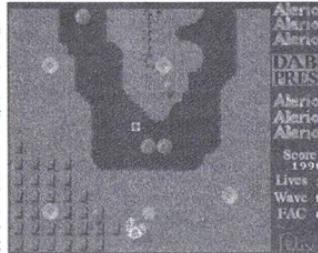
Alerion Archie Arcade Action!

"A welcome return to the traditional shoot-em-up"

Alerion - an eagle without beak or feet is the Arcturian term for impossible and the codename for your mission in this exciting all action game.

Your space-fighter is equipped with revolutionary new equipment, not least a new radar cloaking system which renders you invisible, a holographic targeting system and unlimited fire power. To succeed your task is quite simple ... blow the living daylights out of anything that moves!

You have a bird's eye view of the action, your space-fighter flying over the varied enemy terrain, scrolling effortlessly beneath you. This is done by using the impressive 256 colour, high resolution mode. The game screen is



refreshed 25 times a second by using two 80k screens in a highly innovative fashion, where one is dedicated to update while the other is used solely for display purposes. There is therefore no messy screen swapping. Just as impressive is the use of digital sounds which were sampled from professional sources giving the game an authentic feel.

Alerion is a welcome return to the most popular of all computer games but utilising the power speed, sound (not A305) and superb graphics only available on the world's fastest micro. The

Archimedes.

Alerion costs just £14.95 and is available now!

Archimedes Assembly Language: A Dabhand Guide

The first book specifically written for the Archimedes to provide a complete guide to programming the Archimedes in machine code. Whether you are an Archimedes owner, user, or just interested in the ARM chip this book is the definitive guide. In a massive 368 pages, author Mike Ginn provides a clear, step by step account of using the assembler. Many simple, useful, documented programs provide the practice to illustrate the theory making it ideal for the beginner.

Here's what Risc User said in their latest issue: *"The style of the text throughout the book is easy to read. Good, clear diagrams are used to make explanations easier. I would recommend Archimedes Assembly Language..."* At just £14.95 this Dabhand Guide represents full value for money. A programs disc is available for £9.95 or the two may be purchased for £21.95 when ordered together.

C: A Dabhand Guide

This superb 512 page book is one of the publications of 1988 and provides a step-by-step introduction and programming guide to C on the Archimedes. Beebug had this to say: *"...the book being full of good advice about program design and layout...a very good, reasonably priced introduction to C for the non-specialist."* Book £14.95. £21.95 with disc.

Available from all good bookshops

Free Catalogue and Ordering Details

Write to us, phone us, or send us a mailbox and we will send you, free of charge, our information packed 32 page catalogue giving full details of all our products. We'll also send you details of new books and software.

Send cheques, POs, official orders to the address below, or quote your Access/Visa card number and expiry date. Credit card orders accepted by phone, letter or mailbox. P&P free in UK/BFPO. Elsewhere add £2 or £10 airmail.

**Dabs Press (RU), 76 Gardner Road, Prestwich,
Manchester, M25 7HU. Phone: 061-773-2413
Prestel: 942876210 BT Gold: 72:MAG11596**



ANIMATING ARCHIE (Part 2)

by Lee Calcraft

This month we concentrate on ways to change the aspect of an object.

This program requires a sprite size of 7 for 300 series or 2 for 400 series

In part 1 of this series we looked at the way in which sprites can be created and moved around the screen in various ways. But in any useful animation not only the position of an object, but also its *aspect* must appear to change. The Arc's sprites make this relatively easy to achieve.

In principle, all that we need to do is to create a succession of images of our object, each slightly different from the next, making a separate sprite from each image. To animate our object, we then simply display the sequence of sprites. If the object is to move at the same time, then we display successive images at different positions on the screen.

In practice, the animation process is relatively easy. The difficult bit is generating the images in the first place. Generally speaking there are two approaches. Either you can create the sprites freehand with a drawing package or sprite editor, or you can write a program to generate them automatically. The former route is a painstaking process involving the creation of tens, or even hundreds of images, even for a very brief sequence, and we will confine ourselves, for the moment at least, to the latter.

RECEDING SPHERES

To illustrate the principles of multiple sprite animation, we will take the simple example of a single receding sphere. We will make it appear somewhere in the lower half of the screen, and then recede into the distance. The aspect of the sphere will change in two ways during the sequence: it will obviously get smaller as it recedes, and the way in which it is lit will also change.

We can perform these changes very easily using a modified version of the procedure

PROCsphere from last month. It appears in listing 1 in its new form. To run the program you will need some 50K of sprite space. When it is run it will draw a sequence of 40 shaded spheres in mode 13. After each has been drawn, it is grabbed as a sprite, and re-displayed at the bottom left-hand corner of the screen. If you watch this area you will get some idea of what the animation will look like, except that with the larger spheres, there is a few seconds' interval between images. Once the 40 sprites have been created, they are stored to disc under the name Ssizes40.

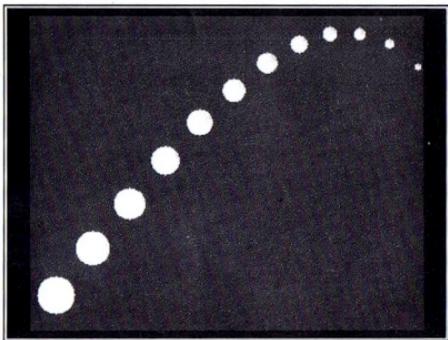
The program has been made quite flexible, and by altering line 100 you can define both the starting radius and the number of spheres to be drawn (set the variable *tot* to one less than the number required). You can also change the mode from 13 to 15 by altering the MODE statement in line 80, and decreasing the variable *xpix%* to 2 in line 90 (remember also to double the sprite space).

Listing 1

```
10 REM >Anim21
20 REM Program Create Sized Spheres
30 REM Version A 0.2
40 REM Author Lee Calcraft
50 REM RISC User September 1988
60 REM Program Subject to Copyright
70 :
80 MODE13: *SNEW
90 xpix% = 4: ypix% = 4
100 rads = 60: tot = 39
110 :
120 FOR no = 0 TO tot
130 rad = rads * (1 - no / (tot + 1))
140 light1 = 90 * (1 - no / tot)
150 PROCsphere (500, 500, rad, light1, 30, 1
8)
160 MOVE 500 - rads, 500 - rads: MOVE 500 + rads, 500 + rads
170 OSCLI ("SGET "+STR$no)
180 CLS
190 PLOT &ED, 0, 0
```

ANIMATING ARCHIE (Part 2)

```
200 NEXT
210 OSCLI("SSAVE Ssizes"+STR$(no))
220 PRINTno; " sprites saved"
230 END
240 =====
250 DEFPROCsphere(X,Y,rad%,L1%,L2%,col%
%) {
260 FOR Y% = rad% TO -rad% STEP -ypix%
270 A% = (SQR(rad%*rad%-Y%*Y%)) DIV xpix%
*xpix%
280 FOR X% = -A% TO A% STEP xpix%
290 P1% = DEG ASN(X%/rad%)
300 P2% = DEG ASN(Y%/rad%)
310 D1 = ABS(P1%-L1%): D2 = ABS(P2%-L2%)
320 C% = 7.99 - SQR(D1*D1+D2*D2)/14 - RND(1)
330 IF C% < 0 THEN C% = 0
340 GCOL0,col%+(C% AND 4)*5.25 TINT(C%
AND 3)*64
350 PLOT69,X%+X%,Y%+Y%:NEXT:NEXT
360 ENDPROC
```



HOW IT WORKS

The program is quite straightforward. It uses a FOR-NEXT loop (line 120) to generate the 40 sprites. The only subtle bit is the assignment of the radius and the light angle in lines 130 and 140. The radius is set to give an impression of linear velocity as the sphere recedes into the distance, and the light position is set so that the sphere is initially lit from the far right, but this shifts towards the centre of the sphere as it recedes, adding to the sense of movement.

Once these calculations have been made, PROCsphere is called, and then the sprite is marked out (in line 160), and saved in line 170. You will notice that the size of the sprite is kept the same each time, although we could have saved sprite space by trimming it each time to fit the sphere (but at the price of complicating subsequent plotting co-ordinates). Finally line 210 saves the whole set of sprites under a filename which reflects the number of sprites generated.

MAKING IT REcede

Now for the easy bit. Listing 2 gives a program to make the sprite appear to recede. In fact it makes two spheres recede to a vanishing point at the centre of the screen. Note that line 60 loads in the 40 sprites, but you only need this if you have cleared the sprite area (or pressed Ctrl-Break etc.) after running listing 1. You can check for the presence of the sprites with *SLIST.

In fact all you need to make the sphere recede is a routine like the following:

```
MODE 13:GCOL 3,0
FOR no% = 0 TO 39
VDU23,27,0,no%
PLOT &ED,X%,Y%
WAIT
PLOT &ED,X%,Y%
NEXT
```

This is similar to the routine employed last month, except that it uses VDU23,27,0,no% to select sprite number no%, rather than *SCHOOSE name. The former offers a considerable speed advantage over the latter, as suggested last month. X% and Y% are the co-ordinates for the plot, and as this routine stands, the sprite will have no translated motion (i.e. it will not move across the screen). As a result, the sense of receding is less powerful, and the eye can interpret its behaviour as shrinking rather than receding. To make sure that the sphere appears to recede, you will need to increment (or decrement) the position

co-ordinates with each new sprite. This is achieved in listing 2 with a pair of sprites (the same one is plotted twice at different places) making them appear to move from either side of the centre of the screen towards a central vanishing point.

Listing 2

```

10 REM >Anim23
20 REM Moves Spheres into distance
30 REM Using Ssize40 sprites
40 :
50 MODE13
60 *SLOAD Ssizes40
70 GCOL 3,0
80 REPEAT
90 Y%-=20:X%-=150
100 X1%-=950:Y1%-=200
110 FOR no%=0 TO 39
120 VDU23,27,0,no%
130 PLOT &ED,X%,Y%
140 PLOT &ED,X1%,Y1%
150 IF no%=0 THEN Z=INKEY(50)
160 WAIT
170 PLOT &ED,X%,Y%
180 PLOT &ED,X1%,Y1%
190 X%+=10:Y%+=10
200 X1%-=10:Y1%+=10
210 NEXT
220 Z=INKEY(100)
230 UNTIL FALSE

```

CAGED SPHERE

Once a sequence of sprites has been created, even if only of a single receding sphere, many different effects can be produced with relative ease. The program in listing 3 makes the sphere bounce around the screen as if it were held in a deep box. The 3D effect works well, with the ball receding into the distance at one moment, then approaching again at the next. A thud is sounded whenever the ball hits one of the box's six invisible walls.

As you can see, this is all accomplished with extreme ease. A single REPEAT loop is used (between lines 90 and 190), and with

each cycle of the loop, three parameters are incremented (or decremented): the sprite number no%, and the X and Y co-ordinates. The three respective increments are i%, j% and k%; and these are negated, and a sound issued, whenever a boundary is reached. It is not possible to draw the bounding box without increasing the complexity of the routine used to detect a bounce. For example the bounce detection routine would need to take into account the current size of the sphere, and so on.

Listing 3

```

10 REM >Anim33
20 REM Caged Sphere
30 REM Using Ssize40 sprites
40 :
50 MODE13:OFF:VDU19,0,24,0,200,64
60 *SLOAD Ssizes40
70 GCOL 3,0:Y%-=500:X%-=500
80 no%=0:i%=1:j%=10:k%=10
90 REPEAT
100 no%+=i%
110 IF X%>1100 OR X%<60 THEN j%=-j%:SOUND 1,-15,70,1
120 IF Y%>700 OR Y%<60 THEN k%=-k%:SOUND 1,-15,70,1
130 IF no%=0 OR no%=39 THEN i%=-i%:SOUND 1,-15,70,1
140 VDU23,27,0,no%
150 PLOT &ED,X%,Y%
160 WAIT
170 PLOT &ED,X%,Y%
180 X%+=j%:Y%+=k%
190 UNTIL FALSE

```

You will see that the sphere is not permitted to go higher than 700 graphics units (line 120). You may like to see what happens if the 700 limit is raised to 900. We will take a closer look at this problem in the next issue. You may also like to experiment with the number of sprites used for the animation.

Next month we will attempt to animate a more complex object.

THE ERROR OF ITS WAYS (2)

Mike Williams concludes his exploration of local error trapping in Basic V.

Last month I gave a short program which formed the quotient of two numbers using local error trapping to check automatically for division by zero. This month I propose to give further examples of local error trapping in action with loops, and with procedures and functions.

Let us now extend last month's example to form a loop which will repeatedly perform the same function as before.

```

100 MODE12: ON ERROR PROCglobal: END
110 LOCAL ERROR
120 REPEAT
130 ON ERROR LOCAL PROCerror
140 INPUT!"Enter two numbers: " X,Y
150 Z=X/Y
160 PRINT X,Y,Z
170 UNTIL FALSE
180 RESTORE ERROR
190 END
200:
210 DEF PROCglobal
220 REPORT:PRINT" at line ";ERL
230 ENDPROC
240:
250 DEF PROCerror
260 IF ERR=18 THEN PRINT"Zero divisor"
270 ENDPROC

```

The two instructions LOCAL ERROR and RESTORE ERROR have been placed around the REPEAT-UNTIL loop, though the RESTORE ERROR in line 180 is strictly redundant. Any exit from a loop (UNTIL, NEXT etc.) will effectively do this anyway, though you may wish to retain the instruction for clarity. The ON ERROR LOCAL statement has been placed inside the loop so that when a *Division by zero* error occurs, the other instructions within the loop are repeated. But there is a flaw, as Escape fails to terminate the loop.

This arises because the local error trap is concerned only with the specific *Division by zero* error. Introducing a flag will allow us to distinguish between the two 'errors'. To achieve this, add or change the following lines to the original program which then works correctly.

```

120 REPEAT:flag=TRUE
130 ON ERROR LOCAL PROCerror
135 IF flag THEN
140 INPUT!"Enter two numbers: " X,Y
150 Z=X/Y
165 ENDIF
170 UNTIL NOT flag
240:
250 DEF PROCerror:flag=TRUE
260 IF ERR=18 THEN PRINT"Zero divisor"
ELSE PROCglobal:flag=FALSE
270 ENDPROC

```

In my view, however, the best solution to the problem is to use a procedure (or function), rewriting the first listing (keeping PROCerror and PROCglobal) as:

```

100 MODE12: ON ERROR PROCglobal: END
120 REPEAT
140 INPUT!"Enter two numbers: " X,Y
150 PROCdivide(X,Y)
170 UNTIL FALSE
190 END
280:
290 DEF PROCdivide(a,b)
300 LOCAL c:LOCAL ERROR
310 ON ERROR LOCAL PROCerror:ENDPROC
320 c=a/b
330 PRINT a,b,c
340 ENDPROC

```

The division is now handled by a separate procedure with its own local error trap. If an error occurs (within the procedure), the procedure exits after printing the error message; if there is no error the two numbers and their quotient are displayed.

Note that in functions and procedures, the LOCAL ERROR instruction must be the last item declared as local at the start of either of these. The RESTORE ERROR instruction, as with loops, is again unnecessary on exit from a procedure or function. Generally, the positioning of the three local error trapping instructions is quite critical, and it is worth making the effort to ensure that you have this correct. I hope my examples (last month and this) will give some help in this respect.



A FAST IMAGE RESCALER

by Stephen Streater

Use this fast image rescaling program by the author of the two-second Mandelbrot to create a variety of interesting effects.

The accompanying program will take any screen image in modes 12, 13 or 15, re-scale it to any proportions, and display it at any point on the screen. There are many uses for such a routine. It could for example be used to produce a screen image catalogue, with a reduced size image of all the screen files on a disc, or it could be used to make an image appear on the screen as a small point, and increase in size; or you could use its scaling qualities to scale a screen picture to fit into any required space. You could even write a small routine to draw with the mouse, using a miniature screen image as a brush. The possibilities are endless.



To get the program working, type it in and save it to disc before running it. When it is run, it will try to load a mode 13 screen called "Screen", and will then run through a display sequence. First the screen will clear, and the image will grow from a point until it is 300 graphics units in size. Then the screen will be filled with 30 small copies of the image, and finally 2000 randomly scaled images will be put on the screen. To give you an idea of speed, the latter sequence runs at around 200 images per second - each individually scaled!

Customising the program to produce other effects is very easy. To see how it is done you need to know that there are two machine code routines, called *copy* and *resize*. The former copies the current screen into workspace

above the program in user RAM. Variables A%, B%, C% and D% are then set giving the x and y graphics co-ordinates of the bottom left-hand corner of the new image, and its width and height. Then *resize* is called, to rescale and display the new image.

A look at lines 170 to 430 of the program, which contain three examples of its use, should make things clearer. As a further illustration, you could insert the following three lines into the program:

```
212 A%=200:B%=200  
214 C%=400:D%=600  
216 CALL resize:END
```

This will place a single image on the screen at 200,200. It will be scaled to a width of 400 and a height of 600 graphics units. To make the image move around, you need to alter A% and B% within a loop, and to make the image change size you need to do the same with C% and D%. Remember to keep the calculations to a minimum in order to avoid flicker. Also, the larger the displayed image the slower the process of drawing it.



The program currently uses mode 13, but works equally well with mode 12 or 15. To change the mode, just alter its assignment in line 100. In its present form the program loads a screen from disc and then copies this into user RAM. You can if you wish load screens directly into user RAM. To do this you will need a screen which has been saved with *SAVE, and

A FAST IMAGE RESCALER



you will need to give the load address as the contents of the label *workspace*. For example:

```
OSCLI("LOAD screen "+STR$~workspace)
```

In this case you will not need to call the *copy* routine, since the screen image will already be in the right place.

```
10 REM          >Rescale
20 REM Program  Image Rescaler
30 REM Version   A 0.09
40 REM Author    Stephen Streater
50 REM RISC User September 1988
60 REM Program  Subject to Copyright
70 :
80 REM (A%,B%)=required position
90 REM (C%,D%)=width and height
100 mode=13:MODE mode:REM 12/13/15
110 PROCconstants
120 FOR pass%=0 TO 2 STEP 2
130 P%:=start%
140 PROCasm(pass%)
150 NEXT
160 SYS "OS_ReadVduVariables",input,ou
tput
170 REM=====
180 *SCREENLOAD screen
190 CALL copy
200 REPEAT
210 CLS:Z%:=INKEY(100)
220 REM----Screen Appear-----
230 FOR Z%=300 TO 10 STEP -8
240 A%:=Z%:B%:=Z%:C%:=400-Z%:D%:=C%
250 WAIT:CALL resize
260 NEXT
270 Z%:=INKEY(200):CLS
280 REM----Multiple images-----
290 FOR J%=0 TO 900 STEP 200
300 FOR Z%:=0 TO 1000 STEP 200
310 A%:=Z%
320 B%:=J%:C%:=190:D%:=190
330 CALL resize
340 NEXT:NEXT
350 Z%:=INKEY(300)
360 REM----Random Sizing-----
370 FOR Z%:=1 TO 2000
380 A%:=RND(1280)-1:B%:=RND(1024)-1
390 C%:=RND(1280)-1:D%:=RND(1024)-1
400 CALL resize
410 NEXT
420 UNTIL FALSE
430 REM=====
440 DEF PROCasm(Z)
```

```
450 [OPT Z
460 .input      EQUD Screenstart
470          EQUD TRUE
480 .output     EQUD 0
490 .copy      \Copy screen into user RAM
500 FN_copy
510 MOV pc, link
520 :
530 .resize \Resize screen
540 FN_box
550 .quit
560 MOV pc, link
570 .workspace
580 ]
590 P%+=80*1024
600 IF mode=15 THEN P%+=80*1024
610 ENDPROC
620 -----
630 DEF PROCconstants
640 IF mode<15 THEN DIM start% 2000+80
*1024 ELSE DIM start% 2000+160*1024
650 Screenstart=148
660 x_low=0:y_low=1
670 x_width=2:y_width=3
680 sp=13:link=14:pc=15
690 IF mode<15 THEN
700 x_trivial=4:shift=6
710 screen_size=80*1024
720 line_size=320:shift_2=2
730 ELSE
740 x_trivial=2:shift=7
750 screen_size=160*1024
760 line_size=640:shift_2=1
770 ENDIF:ENDPROC
780 :
790 DEF FN_copy
800 REM Registers: 0123456789ABC F
810 [OPT Z
820 \Get the screen address
830 ADR R0, input
840 ADR R1, output
850 SWI "OS_ReadVduVariables"
860 \Copy screen to user RAM
870 LDR R0, output
880 ADR R1, workspace
890 MOV R2, #screen_size
900 .copy_1
910 LDMIA R0!, {R3-R12}
920 STMIA R1!, {R3-R12}
930 SUBS R2, R2, #40
940 BNE copy_
950 ]
```



A FAST IMAGE RESCALER

```
960 =0
970 :
980 DEF FN_box
990 REM Registers: 0123456789AC F
1000 LOCAL A%
1010 LOCAL source,dest,temp,sx,sy
1020 LOCAL destnextline,yno,xno
1030 LOCAL horizontal,vertical,offset
1040 temp=0 :source=1
1050 dest=4 :sx=5
1060 sy=6 :destnextline=7
1070 yno=8 :xno=9
1080 horizontal=10 :vertical=11
1090 offset=12
1100 [OPT Z
1110 .box_ \Resample the image
1120 SUB x_width, x_width, #x_trivial
1130 SUB y_width, y_width, #4
1140 CMP x_low, #0
1150 CMPPL y_low, #0
1160 CMPPL x_width, #0
1170 CMPPL y_width, #0
1180 BMI box_end
1190 ADD sx, x_low, x_width
1200 CMP sx, #1280
1210 ADDMI sy, y_low, y_width
1220 CMPMI sy, #1024
1230 BPL box_end
1240 ADD x_width, x_width, #x_trivial
1250 ADD y_width, y_width, #4
1260 \Convert from coords to pixels
1270 MOV x_low, x_low, LSR #shift_2
1280 MOV y_low, y_low, LSR #2
1290 \Calculate the dest address
1300 LDR dest, output
1310 ADD dest, dest, #screen_size
1320 ADD dest, dest, x_low
1330 ADD y_low, y_low, y_width, LSL #
2
1340 ADD y_low, y_low, y_low, LSL #2
1350 SUB dest, dest, y_low, LSL #shif
t
1360 \Calculate the source address
1370 ADR source, workspace
1380 \Get x sampling spacing
1390 MOV sx, #0
1400 MOV temp, #1280<<8
1410 ]
1420 FOR A%=16 TO 0 STEP -1
1430 VDU FN_divide_x(A%)
1440 NEXT
1450 [OPT Z
1460 \Get y sampling spacing
1470 MOV sy, #0
1480 MOV temp, #1024<<8
1490 ]
1500 FOR A%=16 TO 0 STEP -1
1510 VDU FN_divide_y(A%)
1520 NEXT
1530 [OPT Z
1540 \Store dist to nxt line
1550 MOV temp, x_width, LSR #shift_2
1560 RSB destnextline, temp, #line_si
ze
1570 \Zip through sampling it
1580 MOV yno, y_width, LSR #2
1590 MOV vertical, #0
1600 .box
1610 MOV horizontal, #0
1620 MOV xno, x_width, LSR #shift_2
1630 MOV temp, vertical, LSR #8
1640 ADD temp, temp, temp, LSL #2
1650 ADD offset, source, temp, LSL #s
hift
1660 .box_1
1670 LDRB temp, [offset, horizontal, L
SR #8]
1680 STRB temp, [dest], #1
1690 ADD horizontal, horizontal, sx
1700 SUBS xno, xno, #1
1710 BNE box_1
1720 ADD dest, dest, destnextline
1730 ADD vertical, vertical, sy
1740 SUBS yno, yno, #1
1750 BGT box
1760 .box_end
1770 ]
1780 =0
1790 :
1800 DEF FN_divide_x(A%)
1810 [OPT Z
1820 CMP temp, x_width, LSL #A%
1830 ADDPL sx, sx, #1<<A%
1840 SUBPL temp, temp, x_width, LSL #A%
1850 ]
1860 =0
1870 :
1880 DEF FN_divide_y(A%)
1890 [OPT Z
1900 CMP temp, y_width, LSL #A%
1910 ADDPL sy, sy, #1<<A%
1920 SUBPL temp, temp, y_width, LSL #A%
1930 ]
1940 =0
```

If you are eager to rush into print with your Archimedes, take a look at NewsMaster, a desktop publishing package running under the PC emulator.

Mike Williams reports.

LTS Publications has been promoting NewsMaster as a desktop publishing package for the Archimedes for several months now, so we thought we should take a look to see what this software has to offer. On one point you should be clear: NewsMaster runs only under the PC emulator, and the fact that it is supplied as two 3.5" discs is about the only concession (if that) to the fact that it is running on an Archimedes.

There is a glossy 70 page instruction manual, a sheet of clip-board graphics and a Glyph Font chart (a facility whereby the keyboard may be used to input icons in a choice of point sizes).

As with most PC software you are recommended to make backups of the two discs, and then to tailor your working system via a series of prompts. This includes specifying which printer you are going to use out of a very wide choice available. Although it is possible, to some extent, to run the system with a single floppy disc drive, the frequent disc swapping which you are likely to encounter makes the use of a dual-drive system, or a Winchester, highly desirable if not essential. At one point, in following the tutorial guide in the manual, a complete impasse was reached with a single-drive system.

It should also be noted that although NewsMaster is fully icon-driven, it makes no use of the mouse, relying instead on the function keys and cursor keys.

HANDLING TEXT

Clearly NewsMaster is aimed primarily at the production of newsletters at A4 size, and all to a relatively similar format. This consists of a *headline* area of full page width at the top, and one or more columns below. The height of the headline may be varied by whole lines, but there is no choice regarding column widths, margins, gutter size etc. other than the built-in defaults.

A total of 34 fonts is supplied, though that does include variations in style and point size.

For example, a font called *News* is supplied in point sizes 10, 12, 18, 24 and 60, and some of these are available as italic or bold. You cannot independently select font, point size and style. Scrolling through the options on the *font* screen also shows a small sample of the currently selected font. Curiously, one of the fonts referred to in the manual is not included on the disc.

Text may be justified (right or left only), fully justified or centred. It may also be placed against a choice of 30 background patterns. At all times you can view the page as a whole, or zoom in at various levels of magnification. Text can either be typed in through the keyboard, or imported directly from a previously saved ASCII file. Depending on the choice of font, entering text from the keyboard can be painfully slow, as the relevant fonts are loaded from disc, and spaces take just as long as printable characters. Spacing across the line to a particular position (you cannot set tabs or use any other means to speed this up) can really seem painfully slow.

Sections of text may be marked by moving the cursor to the beginning and end of the particular section. Marked text may then be cut or copied to an invisible clip-board for pasting in elsewhere. You can also change the font of selected text. There is also an *Undo* function which allows you to undo any action performed on selected text.

The cursor may be moved around the page with the cursor keys, together with the use of Page Up, Page Down, Home and End (or Copy), and these may also be used in combination with the Ctrl key to move between pages.

INSERTING ARTWORK

NewsMaster also provides facilities for selecting, positioning and editing artwork on a page. A graphics library of some 300 different designs is supplied in a variety of shapes and sizes. Browsing through the library shows each graphic in turn, though this is not particularly fast. According to the manual each graphic has

a meaningful name which is displayed in a menu window for ease of reference, but on my version, the graphics all had coded names making identification impossible other than by seeing the artwork itself on screen. This again causes much time wasting when a particular graphic is sought.

Once a graphic has been selected it may be positioned by placing a box on the page, and scaled by moving the edges of this box. You can also determine how text flows round the piece of artwork, though this is not as fancy as it may at first sound. The choices are for text to flow round both sides, left or right side only, or neither side.

Artwork can also be copied, cropped, and reflected vertically and horizontally, and in conjunction with the graphics library provides considerable versatility. You can also create lines and boxes (frames), and select from a variety of fill patterns. There are no facilities within NewsMaster for creating or editing your own artwork, and no information is given on how graphics might be imported into NewsMaster.

PRINTOUT

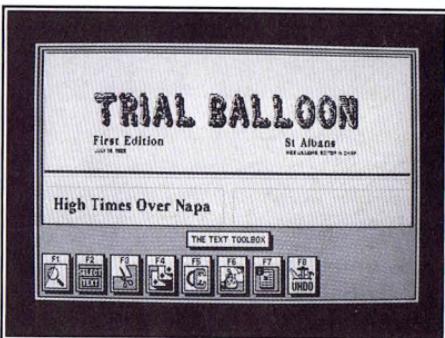
NewsMaster supports a wide range of dot matrix printers, and the quality of output achieved (a Taxan KP 815 was used for this review) is good. The software can also be customised at the outset for your printer.

CONTROLLING THE SYSTEM

When you first enter NewsMaster you have the choice of creating a new page or loading an existing page for further work. All the screens used are basically similar, consisting of a window in the upper two thirds of the screen, which normally contains the current view of the page you are working on, and a row of icons and other reference information at the foot of the screen.

The package as a whole employs a hierarchical menu system, with a set of icons at any time showing the choices available. The icons are generally well designed, and of a good readable size. Each icon is clearly related to one of the function keys (no mouse or pointer here). Escape always returns you to the

next higher menu level. Each menu has a caption on the screen for identification, and help on any icon may be obtained by pressing Alt and the appropriate function key.



CONCLUSIONS

The screen displays in this package are extremely effective, and within its capabilities the system works very well. There is clearly a distinct lack of fine and detailed control over page layouts, and I feel the lack of independent choice of style, point size and font to be a distinct disadvantage. The system, under the PC emulator, is sometimes most annoyingly slow, particularly when extensive disc access is required.

Nevertheless, what NewsMaster does it does well. It really is easy to use and the menu system is easily learnt. If a straightforward newsletter style of approach is what you are after then NewsMaster will do an excellent job, and at an entirely reasonable price, but if your needs are more ambitious (magazines or books), then I fear we have still to wait for the answer as far as the Archimedes is concerned. If it can be done successfully, DTP has to be one of the major applications for the Archimedes in the future.

Product
Supplier

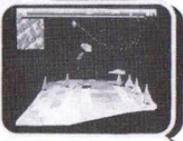
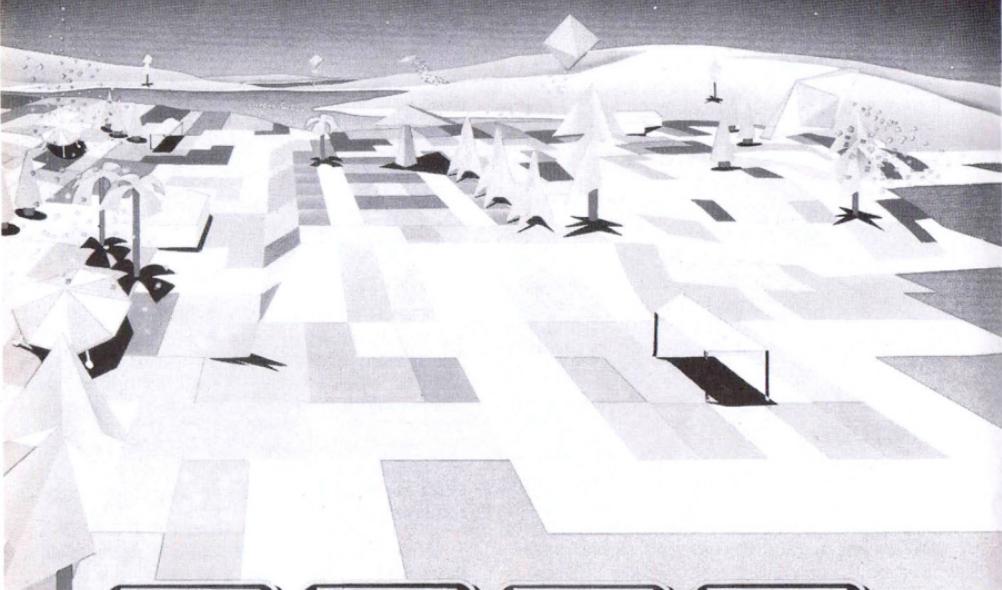
NewsMaster
LTS Publications
Haydon House,
Alcester Road, Studley,
Warwickshire B80 7AN.
Tel. (0386) 792617
£63.25 Inc VAT

Price

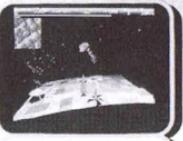
RU

PRIDE
COMPETITION
COULD BE YOUR
GREATEST
ACHIEVEMENT
YET.

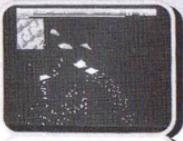
ZARCH



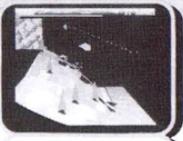
Leaving the Home Base – the Mission commences



A landed Seeder spreads the debilitating red virus



A raging battle against several alien spaceships



Flying fast and low
over a hilly coastline

David Braben is the co-author of the unanimously acclaimed game, *Elite* — probably the world's highest-selling computer game. After months of program development work, David has now harnessed the speed and performance of the Archimedes computer to create another programming masterpiece... *Zarch*.

Zarch is unlike any game you have ever seen before. Even the demonstration program, Lander, supplied on the Archimedes Welcome disc, can only give the most basic idea of the features and fascination that Zarch has in store.



ACORNSOFT

Dept. R2 Regent House, Skinner Lane, Leeds LS7 1AX Telephone: 0532 459453

Please make
all cheques
payable to
"Superior
Software Ltd"



24 HOUR TELEPHONE
ANSWERING SERVICE FOR ORDERS

OUR GUARANTEE

OUR GUARANTEE

- All mail orders are despatched within 24 hours by first-class post.
- Postage and packing is free.
- Faulty cassettes and discs will be replaced immediately.
(This does not affect your statutory rights).

IT'S MORE THAN HEARSAY

David Pilling, author of BEEBUG's new comms software for the Archimedes, reveals some of its features.

When given the opportunity to design a new communications package for a new computer, one pauses to think. Obviously, one could reproduce the typical Beeb comms program like Command. Alternatively, one could clone a PC package like Procomm. But these approaches all belong to the past.

Hearsay, is a true WIMP program; everything is done using windows, pop-up menus, icons and the mouse. In fact, you can dial up a host, move around it, upload and download files and log off without ever touching the keyboard, although keyboard support is of course provided.

Hearsay splits into a number of sections. First, there is the Viewdata terminal. This consists of a window which occupies the left 2/3 of the screen, with a simulated keypad on the right. The window shows frames, and the keypad allows you to enter frame numbers with the mouse and control the functions of the terminal. These include a graphics screen dump, Telesoftware downloading, frame send, frame tagging, and moving frames to and from disc.

Rather than tagging frame numbers, Hearsay stores up to 100 frames in memory, and lets you scan through them in another window, or access directly any individual frame. The Viewdata terminal, also has a powerful editor, with features like block copy/move/ delete, and a pixel editor mode that lets you use the mouse to draw on the screen. In the main terminal mode, the mouse can be used to select Viewdata frame numbers and routes directly.

Next, come the text terminals. Apart from 40 and 80 column dumb Teletypes, Hearsay emulates the DEC VT52, VT102 and VT220. Where Beeb or PC VT100 emulators normally only handle around 50% of the VT100 codes, Hearsay achieves almost 100%. There are even improvements over a real VT100. The main enhancement is a capture buffer, which saves the last 256 lines as they scroll off the top of the screen for instant recall.

Extensive printer support is provided, and as far as possible what appears on the screen can be made to appear on your printer.

For the bulletin board user, the ANSI terminal will be the most interesting one. This gives

access to a multitude of colourful screens from bulletin boards (and Telecom Gold) which provide support for the IBM PC ANSI.SYS driver.



Finally, Hearsay can emulate the Tektronix 4010/14 graphics terminal, with the ability to print/save/load/merge graphics images and to do simple drawing with the mouse. People wishing to transfer files will be pleased to find Kermit and XMODEM, as well as YMODEM, YMODEM Batch, SEAlink and ASCII protocols included as standard.

There is a telephone directory, which allows you to store names, numbers of services and sign-on strings. This ties in with another Hearsay feature, 'status' files. It is possible to save almost all settings and function key definitions in a 'status' file for easy access whenever necessary. Similarly, you can configure Hearsay for your particular type of printer, modem etc. and save the settings in a 'setup' file which is loaded when the program is started.

Drivers for all the popular modems are provided, and there is also the facility to produce your own (no programming involved!) for intelligent modems. All these features, and many more, are fully documented in the 200 page manual that accompanies the software. A number of additional utilities and demos, including the ARChive utility, are also supplied on the Hearsay disc.

Hearsay costs £51.75 (inc VAT) to RISC User and BEEBUG members, or £69 (inc VAT) to non-members.

RU

USING THE PC EMULATOR (2)

Ian Whiting offers some more practical advice on the use of the PC emulator.

The most recent version of the PC Emulator (version 1.09), offers several improvements over earlier versions: it is up to 9% faster, offers more available user RAM (now up to a respectable 614K - remember that MS/DOS has a maximum of 640K), and the installation leaflet now has a page on the Acorn ADFS to MS/DOS file transfer utilities GETFILE and PUTFILE. In addition, a second version of the Emulator, called PC, is included on the disc. This can be run from Arthur with virtually no reconfiguring and leaves 606K of user RAM, and when you press Reset restores your original Archimedes configuration. Unless I need that additional 8K of RAM I now always use PC in place of Emulator.

MS/DOS FILE NAMES

An Archimedes file name can be up to 10 characters long. Full stops are used to denote sub-directories. MS/DOS file names can be up to 11 characters divided into two parts, a prefix of up to 8 characters plus an optional suffix of up to 3 characters. Typically the prefix is the file name and the suffix the file type. A full stop divides the two parts, e.g. STARWARS.BAS would be a Basic program.

A few suffixes are reserved for certain purposes by MS/DOS, including .COM and .EXE for executable programs, and .BAT for batch files. A batch file is similar to an Archimedes EXEC file. Although not mandatory, it is common practice to use the following suffixes:

.BAS	for Basic source programs
.PAS	for Pascal
.COB	for Cobol
.FTN	for Fortran
.OBJ	for object files
.SYS	for system control files
.TXT	for text files
.DOC	for documentation text files
.BAK	for backups
.HLP	for help files

MS/DOS has a directory structure similar to the ADFS, but MS/DOS uses a backslash to denote a sub-directory where the ADFS would

use a full stop; and where the ADFS uses \$ for the root (top) directory, MS/DOS starts the pathname with a back slash. Thus the ADFS pathname:

\$.GAMES .STARWARS
in MS/DOS would be:
\\GAMES\\STARWARS.BAS

The ADFS uses the *DIR command to change directories (e.g. *DIR \$.GAMES). MS/DOS uses the command CD (e.g. CD \\GAMES), or to move to the root directory just enter 'CD \\'.

ADFS pathnames include a disc drive using :0 or :1 (e.g. :0.\$.GAMES), while MS/DOS pathnames can use A: or B: (e.g. A:\\GAMES\\STARWARS.BAS), and A:\\ is the root directory of the disc in drive A. Whereas ADFS uses *MOUNT to switch disc drives, in MS/DOS you just type the disc code, e.g. 'B:'. The MS/DOS system prompt (the equivalent to '*' in ADFS) is usually the disc drive letter, i.e. A> or B>, although this can be changed.

CATALOGUE

The equivalent of the ADFS *CAT command in MS/DOS command is 'DIR', which lists the files one per line. The prefix and suffix are displayed as two separate words. To catalogue a directory, use 'DIR /W' (note the *forward slash*).

DISC FORMATS

A disc may be formatted as a blank data disc or as a system disc. MS/DOS may be booted from any system disc. A system disc may also contain user data. To format a blank 720K data disc, place the MS/DOS system disc in drive A, type:

A:\\FORMAT A:
and when prompted exchange the system disc for a new blank disc. To format a system disc use:
A:\\FORMAT A: /S
instead. This will create two 'hidden' files and the operating system file COMMAND.COM.

Cataloguing the Acorn supplied MS/DOS system disc will show the file COMMAND.COM

(the MS/DOS equivalent of Arthur), and a number of MS/DOS 'external' programs, e.g. DISKCOPY and FORMAT. External programs are similar to Archimedes utility programs, such as DIRCOPY, and are not essential to boot MS/DOS.

FILE COPYING

Use COPY to copy a few files from one disc to another. COPY needs two discs on-line simultaneously. If you have two drives you will use, for example, 'COPY A:FILENAME B:' to copy a file called FILENAME from drive A to drive B.

With single disc drive computers, you must first create a RAM disc to simulate a second drive. You can create a new copy of the MS/DOS disc supplied with the original Emulator package by using DISKCOPY. With this copy in drive A type the following:

```
COPY CON: A:\CONFIG.SYS <Return>
DEVICE=RAMDRIVE.SYS <Return>
<Ctrl-Z> <Return>
```

This creates a new file, or replaces an existing file, called CONFIG.SYS. All subsequent typed lines, until Ctrl-Z, are copied to this file. Now re-boot MS/DOS by pressing the Ctrl, Alt and Delete keys simultaneously to create a 64K RAM disc as drive C (if you create a RAM disc and already have a drive C installed, the RAM drive will become drive D). Files may now be copied from the source disc in drive A to drive C (e.g. 'COPY A:filename C:'), and subsequently, after exchanging discs, copied to the target disc using 'COPY C:filename A:'.

Several files may be copied to drive C in one pass until drive C becomes full. Similar file names may be copied in one go using the 'wild-card' character, the asterisk. For example:

COPY A:*.BAS C:
will copy all files with the suffix '.BAS', and:

COPY C:*.* A:
will copy all files (until the free space in the target drive is exhausted).

After copying the files from the RAM disc to the target disc you must delete the files from the RAM disc before continuing, e.g. 'DEL C:.*'.

Should you have a file larger than 64K you must create a large RAM disc. Re-create CONFIG.SYS as above but add the size, in K, to the end of the DEVICE line, e.g. 'DEVICE=RAMDRIVE.SYS 360' will create a 360K RAM drive. Obviously you cannot exceed the available RAM space and you must leave some RAM for programs to use. If you have a file too large to copy via a RAM disc then DISKCOPY the entire disc to a new disc and delete any unwanted files.

To remove the RAM disc, re-boot with a normal copy of the MS/DOS disc, or delete the CONFIG.SYS file using the command 'DEL A:\CONFIG.SYS'. All files left in the RAM disc before re-booting will be lost.

ENVIRONMENT

In MS/DOS you alter the configuration of the system by changing the \CONFIG.SYS file and then re-boot with <Ctrl-Alt-Delete>. The RAM disc example above is one such change.

Other environmental changes can be made immediately without re-booting. The most common is the PATH command. This lists the order in which MS/DOS searches sub-directories for required programs. Several directories can be specified in one PATH command, e.g. 'PATH A:\;A:\DOS', by separating each directory name with a *semi-colon*.

Whenever you enter a program name, e.g. 'FORMAT', MS/DOS checks first to see if it is an 'internal' command (e.g. DIR and CD are internal commands), and then looks in the current directory for a .COM file (e.g. FORMAT.COM) of that name, then a .EXE file of that name, and finally a .BAT file of that name. If no match is found it then looks in the first directory listed in the PATH command for a .COM, .EXE or .BAT file. If no match is found, it checks the next directory, and so on. If no match is found after checking all directories in the PATH command, the message 'Bad command or file name' is displayed.

That's all on the PC emulator for now. Let me know if there are any particular topics you would wish to see covered.

Archimedes Visuals

This month: A painting program complete with airbrush in just 20 lines - plus 3D cylinder and cone drawing routines

SPRAYCAN

By Julian Mudd

This remarkable program implements a complete mouse-driven mode 13 art package including variable size airbrush, flood fill, full-colour palette, and the ability to save and load screens - all in 20 lines of code. Just type it in, and away you go.

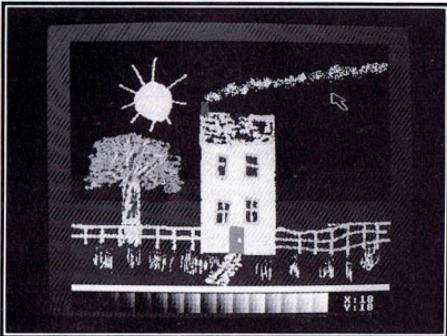
	Drawing area	Below the line
Select	Spray	Select colour
Menu	Flood fill	Inc spray width
Adjust	Save Screen	Inc spray height

Mouse controls

When you run the program it offers the chance to load a picture. If you respond with "Y" it will try to load a mode 13 screen called "SpryPic". Once on the drawing screen, use the mouse as indicated in the table. To adjust the spray size use the menu and adjust buttons with the pointer below the drawing line. The brush size is constantly displayed, and cycles around from 63 back to 0.

```
10 REM >Spraycan
20 REM Program Airbrush Painter
30 REM Version A 0.3
40 REM Author J.H.Mudd
50 REM RISC User September 1988
60 REM Program Subject to Copyright
70 :
80 MODE 13:OFF:VDU19,0,24,96,96,96
90 PRINTTAB(8,30)"Load Pic (Y/*) ?"
100 IF GET=89 THEN *SCREENLOAD SpryPic
110 FOR C%=0 TO 255
120 GCOL C% DIV 4 TINT C% MOD 4<<6
130 RECTANGLE FILL C% MOD 64*16,C% DIV
64*20,12,16
140 NEXT
150 B%:=7:I%:=1:J%:=1:X%:=1023:Y%:=79
160 *POINTER
170 REPEAT
180 IF Y%>(87+J%) THEN
190 IF B% AND 4 A%:=RND(360):POINT X%+R
ND(I%)*COSRADA%-4,Y%+RND(J%)*SINRADA%+4
200 IF B% AND 2 GCOL 128+POINT(X%,Y%)
```

```
TINT TINT(X%,Y%):FILL X%,Y%
210 IF B% AND 1 THEN *SCREENSAVE SpryP
ic
220 ENDIF
230 IF Y%<80 THEN
240 IF B% AND 4 IF X%<1024 GCOL POINT(
X%,Y%) TINT TINT(X%,Y%):RECTANGLE FILL 0
,84,1279,20
250 IF B% AND 2 I%=(I%+1) MOD 64:K%:=IN
KEY(8):PRINTTAB(34,30)"X:";I%;" "
260 IF B% AND 1 J%=(J%+1) MOD 64:K%:=IN
KEY(8):PRINTTAB(34,31)"Y:";J%;" ";
270 ENDIF
280 MOUSE X%,Y%,B%
290 UNTIL FALSE
```

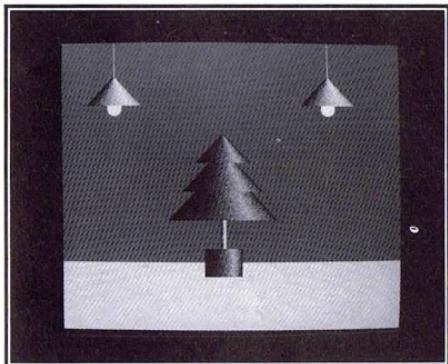


3D CONES AND CYLINDERS

By Lee Calcraft

This visual item features procedures for drawing 3D cones and cylinders. They are both based on the 3D sphere-drawing routine by Mark Davis published last month, and use the same dithering algorithm to shade the objects. They are accompanied by two procedures PROClamp and PROCtree to illustrate their use. When you run the complete program, the image produced has a really good 3D feel to it, even though this may not be apparent in our screen shot. As the program stands it uses mode 13. To use mode 15, change the MODE statement in line 80, and alter the value of xpix% in line 100 to 2. And for an interesting effect, try it in mode 13 with ypix%=8.

Both procedures have seven virtually identical parameters: the first two give the base co-ordinates of the object, the third and fourth its height and radius, the sixth the angle of illumination (in degrees), and the seventh the colour number to be used, where the same restrictions apply as last month. The fifth parameter differs in effect from cone to cylinder. In the former case it specifies whether the cone's apex is above or below its base (TRUE or FALSE respectively); and in the latter whether the cylinder is vertical (TRUE) or horizontal (FALSE).



```

10 REM          >ConeCyl
20 REM Program  Cones & Cylinders
30 REM Version  A 0.5
40 REM Author   Lee Craft
50 REM RISC User September 1988
60 REM Program  Subject to Copyright
70 :
80 MODE13:GCOL144 TINT240:CLG:GCOL42
90 RECTANGLE FILL 0,0,1279,250
100 xpix%=4:ypix%=4
110 PROClamp(200)
120 PROClamp(1000)
130 PROCTree
140 END
150 :
160 DEFPROClamp(hor)
170 PROCyl(hor,900,123,2,TRUE,30,6)
180 GCOL 0,63:CIRCLE FILL hor,790,25
190 PROCon(hor,800,100,100,TRUE,30,8)
200 ENDPROC

```

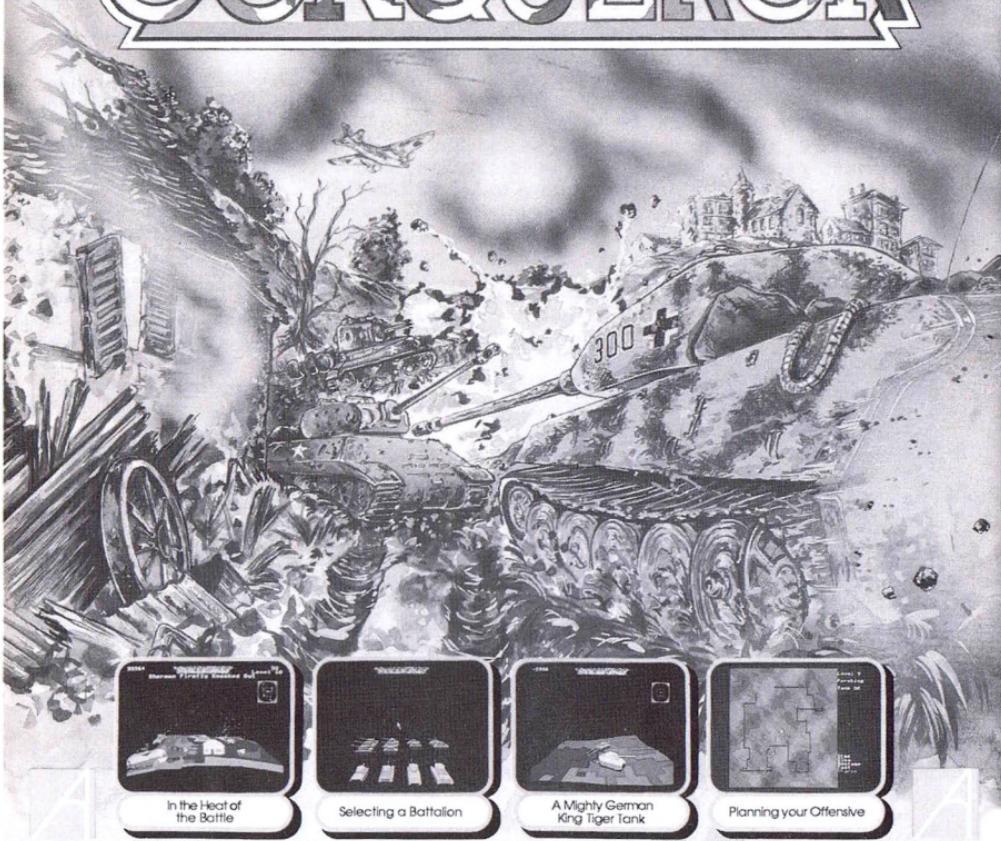
```

210 :
220 DEFPROCTree
230 PROCyl(600,200,96,70,TRUE,30,2)
240 PROCyl(600,300,100,10,TRUE,30,6)
250 PROCon(600,400,200,200,TRUE,30,4)
260 PROCon(600,500,150,150,TRUE,30,4)
270 PROCon(600,600,100,100,TRUE,30,4)
280 ENDPROC
290 =====
300 DEFPROCon(X,Y,ht%,rad%,upright,L1
%,col%)
310 FOR Y%=ht% TO 0 STEP -ypix%
320 IF upright THEN YY%=Y% ELSE YY%=ht
%-Y%
330 A%=(rad%DIV xpix%)*xpix%*(ht%-YY%
/ht%)
340 IF A%>0 THEN
350 FOR X%=-A% TO A% STEP xpix%
360 P1%=DEG ASN(X%/A%)
370 D1=ABS(P1%-L1%)
380 C%=7.99-D1/14-RND(1)
390 IF C%<0 THEN C%=0
400 GCOL0,col%+(C% AND 4)*5.25 TINT(C%
AND 3)*64
410 PLOT69,X+X%,Y+Y%:NEXT
420 ENDIF:NEXT:ENDPROC
430 =====
440 DEFPROCcyl(X,Y,ht%,rad%,vert,L1%,c
ol%)
450 ystep%=ypix%:xstep%=xpix%
460 IF NOT vert THEN SWAP ystep%,xstep%
470 FOR Y%=0 TO ht% STEP ystep%
480 A%=(rad%DIV xstep%)*xstep%
490 FOR X%=-A% TO A% STEP xstep%
500 P1%=DEG ASN(X%/rad%)
510 D1=ABS(P1%-L1%)
520 C%=7.99-D1/14-RND(1)
530 IF C%<0 THEN C%=0
540 GCOL0,col%+(C% AND 4)*5.25 TINT(C%
AND 3)*64
550 IF vert THEN PLOT69,X+X%,Y+Y% ELSE
PLOT 69,X+Y%,Y+X%
560 NEXT:NEXT:ENDPROC
570 =====

```

Using these two procedures with the sphere procedure from the last issue it is possible to create some very effective images with very little code. This month's magazine disc features a routine to draw a snooker table.

CONQUEROR



In the Heat of the Battle



Selecting a Battalion



A Mighty German King Tiger Tank



Planning your Offensive

CONQUEROR—The Software Revolution Rolls On

Step into the tank of your choice and experience the thrill of controlling one of the most powerful land vehicles in the world. Masterfully designed and written by Jonathan Griffiths, with graphics routines by David Braben, this astounding game will fascinate you as your skills of driving, firing and strategical thinking develop and improve.

Conqueror takes place in a 3-dimensional world of roads, hills, villages, rivers and woodland—which only the Archimedes computer can portray with such reality. Set against this environment is a range of tanks: each authentically reproduced taking into account its speed, armour and firing capabilities.

Three distinct types of game are supplied in Conqueror, to cater for everyone from the novice to the expert:—

Game type 1 (Arcade) is a simple arcade-style game enabling you to develop your control of driving and firing.

Game type 2 (Attrition) puts you in command of a small group of tanks. You must initiate yourself with the principles of battle strategy in addition to controlling the actions of an individual tank.

Game type 3 (Strategy) is a full battle simulation with up to 16 tanks per side from American, German or Russian forces. You can call upon indirect fire to knock out the enemy, and use spotter planes to locate the opposing forces as they attempt to take the strategic objectives.

This game is well-documented with extensive information provided on all of the 12 types of tank involved. Now's the time to see if you can become...The Conqueror.

**SUPERIOR
SOFTWARE**

ACORN SOFTWARE

Dept. Q3, Regent House, Skinner Lane, Leeds LS7 1AX. Telephone: 0532 459453

Please make
all cheques
payable to
"Superior
Software Ltd."



24 HOUR TELEPHONE
ANSWERING SERVICE FOR ORDERS



OUR GUARANTEE

- All mail orders are despatched within 24 hours by first-class post.
- Postage and packing is free.
- Faulty cassettes and discs will be replaced immediately.

(This does not affect your statutory rights.)

AN ADFS FILE-FIND UTILITY

by Lee Calcraft

Use this short utility to find lost files on your ADFS disc, or to catalogue a whole disc.

The short accompanying program implements a full-feature file finding utility, when used in conjunction with three procedures developed in last month's article "Reading from the ADFS". To make it work you will need to append lines 390-820 of last month's program to this month's listing (i.e. you need the definitions of PROCreaddisc, PROCgetname and PROCgetit).

When you run the program a filename and start directory will be requested. When giving the former, you may use wildcards # and * to denote a single character or any number of characters respectively. When supplying a starting directory, an empty Return is taken to imply the root directory. If you need a printout, then execute Ctrl-B immediately before pressing Return on the start directory. When the program terminates, use Ctrl-L to produce a form feed, and Ctrl-C to turn the printer off. To obtain a listing of all the files on a disc, enter an asterisk (*), and press Return twice.

The program revolves around the procedure PROCsearch, which is used recursively. It searches the list of objects in a given directory until it finds a match, or until it finds a sub-directory. In the latter case it calls itself to search the sub-directory. It continues in this way until it reaches the last object in the directory currently under scrutiny. It then moves up one level to continue with the parent directory, and so on until the task is complete.

Note: The size of the two buffers **discbuff** and **dirname** have been made substantially greater than in last month's program to cater for greater numbers of objects and more deeply nested directory structures.

If you are using Econet, you should again replace the four occurrences of the number 77 by 147 (lines 100 and 110).

```
10 REM          >ADFSFind
20 REM Program   Find File Utility
30 REM Version   A 0.8
```

```
RUN
Filename? *prog*
Start Directory? $.Main
$.Main.MyProg1
$.Main.HisProgram
$.Main.Sub.PROGRAM4
>
RUN
Filename? #ROG*
Start Directory?
$.Progs1
$.Frogger
$.Main.Sub.PROGRAM4
$.Zoo.FrogProg

40 REM Author      Lee Calcraft
50 REM RISC User   September 1988
60 REM Program     Subject to Copyright
70 :
80 DIM discbuff &2000
90 DIM dirname &1000,wildcard &20
100 DIM dir$(77),dir(77,1)
110 DIM file$(77),file(77,1)
120 INPUT "Filename ",name$
130 INPUT "Start Directory ",dir$
140 IF dir$="" dir$="$"
150 PROCsearch(name$,dir$)
160 END
170 :
180 DEFPROCsearch(name$,path$)
190 LOCAL n
200 PROCreaddisc(path$,name$)
210 IF filecount>0 THEN
220 FOR N=0 TO filecount-1
230 PRINTpath$;".";file$(N)
240 NEXT
250 ENDIF
260 PROCreaddisc(path$,"*")
270 IF dircount>0 THEN
280 FOR n=0 TO dircount-1
290 PROCsearch(name$,path$+"."+dir$(n)
)
300 NEXT
310 ENDIF
320 PROCreaddisc(path$,"*")
330 ENDPROC
340 :
```

GammaPlot, a business graphics package, is Minerva's latest release for the Archimedes. Mike Williams previews this new software.

GammaPlot is a complete business graphics system, and consists of a single disc and accompanying manual. The review copy was a pre-release version, but I am pleased to say that the software performed faultlessly, and the early version of the manual is a definite improvement on some of Minerva's earlier efforts.

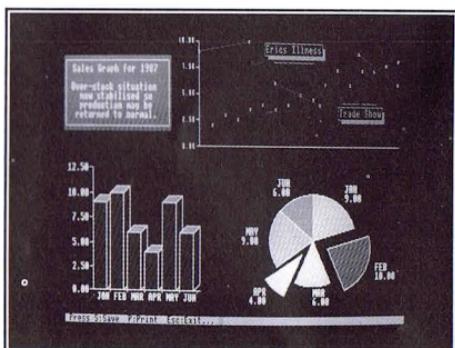
The purpose of GammaPlot can be stated very simply: it is to allow the user to enter data which may then be displayed in a variety of different graph formats. Within that objective, GammaPlot has much to offer, and the results are both colourful and highly effective.

I have been critical of Minerva in the past (see the review of System Delta in RISC User Issue 3) for indulging themselves in colourful but difficult to fathom icons, and displays that are more confusing than helpful. With GammaPlot I have nothing but praise in this respect. The screens are models of clarity, and all options are displayed in text form. Furthermore, the package is to a large extent mouse driven, but the keyboard often provides an alternative, and the dual control approach makes for greater convenience in use, not less.

MAIN MENU

Once past Minerva's slick animated title screen, you are presented with GammaPlot's main menu with 16 choices including the entry of star commands. Any option may be selected by moving a scroll bar to the required option and pressing Return, or by typing the initial letter of the option. So for star commands you just type '*' and the command.

There are options for loading and saving graphs, and for importing an ASCII file. Several example files are contained on the disc supplied, and the so-called 'experimental' section of the manual uses these in a tutorial mode. Whether or not you load in existing data, your next step is almost certainly to select the data entry option (by pressing Escape).



DATA ENTRY SCREEN

This screen is in three sections. The lower half displays the currently entered data (if any), and is used for both entering new data, and editing existing data. Each entry in this table comprises a *label*, X and Y values and two attributes called *colour* and *high*. The X and Y columns are used to enter data (frequencies for example) for single or bi-variate distributions. The *label* field provides identifying labels on appropriate graphs. The *colour* attribute allows the user to specify the colour for any particular entry, while the *high* attribute allows one or more entries to be highlighted. In a pie chart, for example, highlighted segments are 'pulled out' slightly in comparison with the rest.

The data entries can be scrolled up and down as required, new lines can be inserted and existing lines deleted. Clicking on an existing entry displays an entry box with the old value for modification. If you wish to see more data, then clicking on a *display* option converts the whole screen into a data display with a one line summary of other parameters.

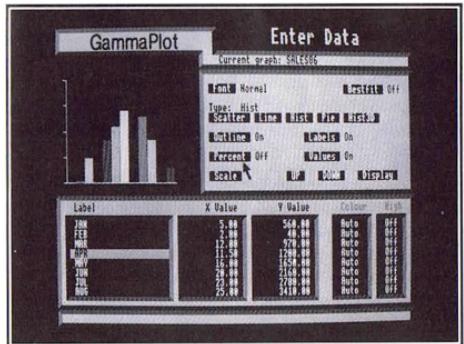
The upper half of the default data entry display shows to the left, a small version of the current graph of your data. It is fascinating to see this change in real time as you enter or change the data from which it is derived. The

PLOTTING WITH MINERVA

remainder of the screen allows you to select a variety of options which control the way in which the data is displayed.

Foremost among these, of course, is the choice of graph itself (Scatter, Line, Histogram, Pie Chart, or 3D Histogram). Four other features may be toggled on or off. These are *outline, labels, percent and values*. With outline selected bars or pie chart segments have a white outline. The percent and values options are mutually exclusive, but both may be switched off.

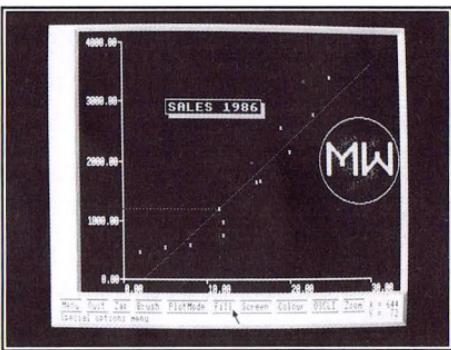
Text can be displayed in normal, thin or small fonts, and you can specify a line of best fit to be displayed automatically for line and scatter graphs. The user can also specify the minimum, maximum and step sizes for both the X and Y variables, or leave this to the software. All the features of this screen work well in practice, and indeed it is quite an enjoyable experience editing the data and marshalling all the attributes for a most effective graphic display.



DISPLAYING GRAPHS

Back to the main menu and select D (for Display Graph), and the graph we have selected and designed is displayed at full screen size. A useful feature here is the window option which allows a box to be placed around a graph. The box may then be moved and re-sized and once confirmed, the graph is automatically re-displayed. Using this feature

several graphs may be displayed on the same screen by reducing them and positioning them in different parts of the screen. Once a graph display has been completed to your satisfaction it may be saved for customisation. It can, of course, be printed out at this stage too.



CUSTOMISING GRAPH DISPLAYS

Return again to the main menu and select the *cutaneous* option. This allows a basic graph display to be embellished in a variety of ways. One point caused some temporary difficulty at this stage. I had saved two graph displays to one of my own discs for customising. However, when you select *customise* from the main menu, GammaPlot loads in a new module, so the GammaPlot disc must be correctly mounted before this can proceed. From the *customise* menu the only way of mounting my data disc was to choose the *other* option, and from within that the OSCLI option to mount the correct disc. In my view the user should not have to find obscure ways of typing in star commands in order to swap discs. The software could be written to make this aspect simpler.

GammaPlot provides almost a complete art package for customising graphs with a good number of options: geometrical shapes, brushes, fill, text, colour, boxes and more. However, as with many art packages, care is needed, as it is much easier to add features to the display than it is to remove them later. There is also a facility for setting up a slide show.

Continued on page 46

Ace Computing

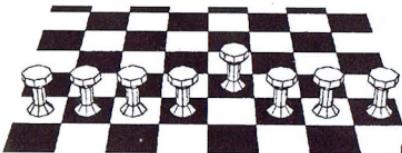
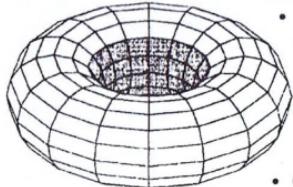


Euclid

is a fully programmable 3-D modelling and animation system. It allows instant visualisation of scenes in 3-D which you can then view from a variety of positions and angles, with camera control like a TV studio. Objects can be moved around and constructed live on screen.

Euclid

- Typical drawing times of 1/5th second for full screen
- Intersecting and concave surface polygons
- Line or surface drawing
- Full hidden surface removal
- Instant scene and object transformations
- Works in any graphics mode
- Multiple use of solid definitions in one scene
- Orthogonal or perspective projections
- Conversion to and from DXF format
- Supports a number of colour printers at full resolution



Cost: £45 (including VAT and P&P)



Order Form

Please send me _____ copy/copies of Euclid.

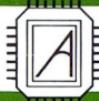
I enclose a cheque/PO for £ _____ (made payable to Ace Computing).

Name: _____

Address: _____

Date: _____ Signed: _____

Please complete and return to: Ace Computing, 27 Victoria Road, Cambridge CB4 3BW



INTRODUCING ARM ASSEMBLER (5)

by Lee Calcraft

This Month: More on LDR and STR Multiple Load and Store and a Fast Block Move Routine

Last month we looked at the LDR and STR instructions, used for loading data from memory into the ARM's registers, and vice versa. These instructions can take one of three addressing modes, and last month we looked at pre-indexed indirect addressing. To recap, the following instruction:

LDR R0, [R1, #8]

uses this addressing mode to load register R0 with the contents of RAM at address R1+8. The index (in this case the value 8) is added to the contents of R1 *before* the load occurs: hence the term *pre-indexed* addressing.

PC RELATIVE ADDRESSING

The two remaining forms of addressing, which again apply equally to LDR and STR operations, are *post-indexed addressing* and *PC relative addressing* (PC=program counter). We will look at the latter first. PC relative addressing, available only on LDR and STR instructions, takes the following form:

LDR <Rn>, <expression>

Its effect is to load register Rn with the contents of RAM at the address given by *expression*. Typically the expression will be the name of a label, but could equally be any expression known to Basic, providing that it evaluates to an address at the time of assembly. For example:

```
LDR R0,data
LDR R1,data+4
.
.
.data
EQUD &FF00FF00
EQUD &00FF00FF
```

The way in which the assembler handles this is to calculate an offset to the current position of the instruction (as given by the contents of the program counter), and use this rather than the absolute address, in its machine code instruction. You may remember from last month that there is not enough room in any 32-bit load or store instruction to hold a complete address. For the same reason the degree of

offset is also limited. Its range is -4095 to +4095, and the assembler will give an error message if an attempt is made to use a greater offset.

One important consequence of this form of addressing is that it permits the programmer to write so-called *position-independent code*. This is code which can be relocated to any word-aligned address without the need for reassembly. This is highly desirable in any case, and absolutely essential in the case of relocatable modules, where the user has no control over the load address.

THE ADR DIRECTIVE

While on the subject of position independence, it is worth digressing for a moment to compare the PC relative LDR instruction to the assembler directive ADR, introduced in part 3 of the series. The ADR directive assembles to an ADD or a SUB instruction involving the PC which again confers position independence. But ADR is used to load a given register with an *address*, while LDR is used to load *data* from the specified address. To clarify the difference, consider the following:

```
LDR R0,data
ADR R1,data
.
.
.data
EQUD &00FF00FF
```

After the two instructions have executed, register R0 will contain the word at location *data* (i.e. &00FF00FF), while R1 will contain the address at which *data* appears in the assembly. The mnemonic *ADR* should serve as a reminder that this directive loads *addresses*, not data.

POST-INDEXED INDIRECT ADDRESSING

With pre-indexed addressing, the index is added to the load or save address *before* memory is accessed. With post-indexed addressing the index is added to the address



after memory access. Here is an example:

```
LDR R0, [R1], #4
```

This instruction will load register R0 with the data at the address held in R1. It will then increment the contents of R1 by 4, so that it points to the next 32-bit word in RAM. Alternatively,

```
LDR R0, [R1], R2
```

will load R0 with the contents of the address held in R1, and then add R2 to R1. This phenomenon, in which the load or store address is automatically updated after memory access, is called *write-back*. It allows you to step through memory with great economy. For example, consider the three instructions:

```
LDR R0, [R5], #4
```

```
LDR R1, [R5], #4
```

```
LDR R2, [R5], #4
```

The first loads R0 with the word at the address given by R5, the next loads R1 with the next word (since R5 has been incremented by 4 in the previous instruction), and so on. We could equally well have decremented R5 using:

```
LDR R0, [R5], #-4
```

Write-back is a powerful feature, and it is possible for the programmer to request it in other memory access instructions by using the "!" symbol. For example, in the following pre-indexed instruction:

```
LDR R0, [R5, #16]
```

register R0 is loaded with the contents of the address found by adding 16 to the contents of R5. We can force write-back by adding an exclamation mark:

```
LDR R0, [R5, #16]!
```

With this variant, the load will take place at address R5+16 as before, but afterwards this load address will be written back to R5.

Since there is no time overhead with the introduction of write-back, its use can considerably reduce execution times by avoiding repeated additions when accessing a sequence of memory locations. But there is an even faster way of performing multiple register loads and saves using two special instructions.

MULTIPLE LOADS AND SAVES

The two ARM instructions LDM and STM will load and store multiple registers. The

syntax is a little complex because of the flexibility which the ARM instruction set provides. We will begin with an example:

```
LDMIA R12, {R0, R1, R6-R9}
```

This instruction will load registers R0, R1, R6, R7, R8, and R9 from RAM starting at a base address given by the contents of R12. In this form there is no write-back, though 4 is added to the address held in R12 after each load. If we add an exclamation mark after the base register, the final address will be written back to R12 so that it points to the next free location in the table of data:

```
LDMIA R12!, {R0, R1, R6-R9}
```

The mnemonic's two suffixes I and A indicate that the contents of R12 should be *incremented* (rather than decremented), and that the increment should occur *after* (rather than before) the load is made. There are thus four possible permutations for both load and store operations:

LDMIA	STMIA	Increment After
LDMIB	STMIB	Increment Before
LDMDA	STMADA	Decrement After
LDMDB	STMDB	Decrement Before

For most applications (apart from stack operations, to be covered next month) the first of these options, Increment After, is used. As with all ARM instructions, condition suffixes may also be used, resulting in quite complex looking instructions. For example:

```
LDMIANE R12!, {R0, R1, R5-R9}
```

As you can see, the condition suffixes appear at the end of the mnemonic string.

SCREEN MOVING

It is an extremely easy matter to write very fast bulk move routines in ARM assembler using the multiple load and store instructions with write-back. As an example, take a look at the following:

```
LDR R8, source
LDR R9, dest
LDR R10, size
ADD R10, R10, R8
.loop
LDMIA R8!, {R0-R7}
STMIA R9!, {R0-R7}
CMP R8, R10
```



```
BLO loop
.
.
.source EQUD sourceaddress
.dest EQUD destinationaddress
.size EQUD screensize
```

This very compact routine transfers 32 *bytes* of RAM at each gulp (because it uses eight 32-bit registers), and will move 80K of data in around one hundredth of a second.

It begins by loading the source and destination addresses, and size of block to be moved into registers R8, R9 and R10 (note the use of LDR with PC relative addressing). It then calculates the final load address by adding the source address to the size of block. And then it is off, using multiple loads and stores (in registers R0 to R7) with write-back, to transfer data at an incredible rate. The only proviso is that if the two areas (source and destination) overlap, then the destination must be *lower* in RAM by at least 32 bytes. If this is not the case, then the routine should be altered to use LDMDA and STMDA, and the start addresses adjusted accordingly.

The program in listing 1 uses a similar routine to copy the contents of shadow RAM into the main screen area. At the first press of the space bar, it clears the current screen, and at the second it copies the shadow screen across. As the program stands, it uses mode 12, and expects to find a screen named SCREEN in a directory called SCREENS. For the program to work correctly, you will need 160K of screen RAM since the main and shadow screens require 80K each. However, all the screen data are legally obtained from the procedure PROCscrnparams, and you should find that by changing the mode statement in line 80, it will work with any size screen, providing that you have configured enough screen RAM, and that you have an appropriate image in the SCREENS directory.

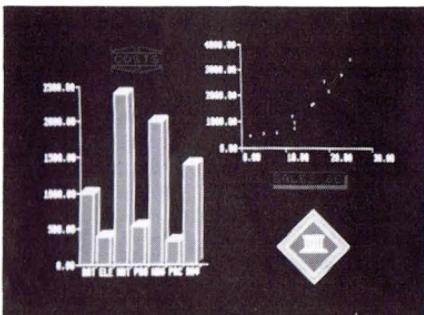
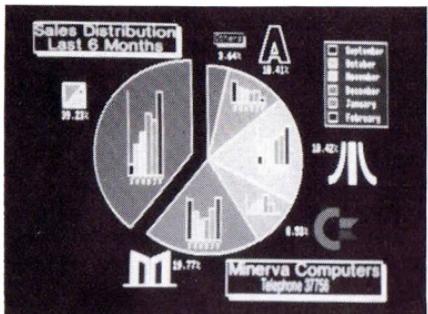
I should add however, that this program does not copy across the colour palette to the new screen, so if a special palette has been used with your screen, this will be ignored.

```
10 REM >Assem5
20 REM Program Screen Mover
30 REM Version A 0.8
40 REM Author Lee Calcraft
50 REM RISC User September 1988
60 REM Program Subject to copyright
70 :
80 MODE12
90 DIM buff &30,space &1000
100 PROCscrnparams
110 PROCassemble
120 OSCLI("LOAD SCREENS.SCREEN "+STR$(base+size))
130 REPEAT
140 CALL blockmove
150 IF GET:CLS:IF GET
160 UNTIL FALSE
170 :
180 DEFPROCassemble
190 FOR pass=0 TO 1
200 P%=space
210 [
220 OPT pass*3
230 .blockmove
240 ADR R12,data
250 LDMIA R12!,{R8-R10}
260 ADD R10,R10,R8
270 .loop
280 LDMIA R8!,{R0-R7}
290 STMIA R9!,{R0-R7}
300 CMP R8,R10
310 BLO loop
320 MOV PC,R14; Return
330 :
340 .data
350 EQUD base+size
360 EQUD base
370 EQUD size
380 ]:NEXT
390 ENDPROC
400 :
410 DEFPROCscrnparams
420 !buf=148:! (buf+4)=7
430 !(buf+8)=150!: (buf+12)==1
440 SYS "OS_ReadVduVariables" ,buff,buff+&10
450 base=buf!&10:size=buf!&14
460 ENDPROC
```

Next month we will be covering the postponed subject of stacks and subroutines.

GAMMAPLOT

Presentation Graphics with a full Art package
to unleash the power of the Archimedes



Seeing is believing!

Gammaplot is an extensive chart / graph plotting program with a full art package included. It allows even a novice to produce stunning charts and drawings on one screen. Several charts may be placed on one sheet, in any position with text or drawings to make the data easily understood. GammaPlot supports six main graph and chart types of line, scatter, pie, histogram, 3D histogram and text only charts. Segments may be highlighted or percentages, values and labels switched on. Statistics and a full Slide Show are also included.

Choose from the text styles supplied or any of the native Archimedes 'pretty fonts' or even define your own, and any of these can be enlarged or reduced. Various shapes, patterns and borders are supplied with a full 256 colour range and a flood fill option. A full zoom facility allows editing of individual pixels. Various pen types and spray cans are provided to draw pictures or add signatures. Block options include move, copy and squash.

Import digitised pictures or company logos. Print facilities include colour output. Data entry is via the keyboard or any Delta, Gamma or Sigma file or most other graphics programs. Screens may be saved and merged automatically with wordprocessors like 1st Word Plus.

So extensive is GammaPlot that recent presentations to educational sectors show that many schools are buying GammaPlot for the art package alone!

AFFORDABLE



EXCELLENCE



MINERVA

COMPUTER SOFTWARE AND SYSTEMS SPECIALISTS

69 SIDWELL STREET · EXETER EX4 6PH · TELEPHONE EXETER 0392 437756



RISC USER TOOLBOX (Part 3)

David Spencer adds a disc sector editor to the RISC User Toolbox.

This month's addition to the Toolbox provides two related commands that allow the individual sectors of a disc to be edited directly. There are many useful tasks that such editing can perform. For example, in some cases it might be easier to change a single byte in a file by editing the actual disc, rather than loading the file into memory, altering it, and saving it again. Other possible uses include recovering lost files, salvaging damaged discs, and implementing a simple form of disc protection. While the ability to directly edit the sectors on a disc is very useful, it can also be very dangerous. If certain areas of a disc, for example the sectors containing the free space map, are edited at random, then the entire disc contents could be lost. The moral of this is always backup a disc before editing it, especially if you are not quite sure of what you are doing.

As with the previous additions, the new lines given in listing 1 should be added to the existing Toolbox source code (in other words the original listing plus last month's additional lines). Before starting this, ensure that the original program has not been renumbered in any way. Once all the new lines have been entered, the source code should be saved, using a different filename to the original, just in case anything goes wrong. As before, running the program will assemble the Toolbox module and save it to disc. This can then be loaded in the way described in part 1. This month's RISC User disc contains the complete source code for the Toolbox so far.

The new commands both perform a similar function - invoking the sector editor. They differ in how the start sector is specified. The first of these is:

*DEDIT [<drive>] <disc address>

This starts editing at the specified disc address. Disc addresses are similar to memory addresses, but refer to the bytes on a disc instead of memory locations. Address 0 refers to the first byte of the first sector of track zero. An optional drive number can be specified, either as a digit in the range 0 to 7, or a letter A to H. This can be preceded by a colon if desired. If a drive number is not specified, then the default drive is used. This is the drive set with the *DRIVE command, and is not necessarily the drive that is currently mounted.

The syntax of the second command is:

*DEDITT [<drive>] <head> <track> <sector>

This will invoke the editor at the specified sector,

track and head. For 800K D format discs, the track number is in the range 0 to 79, with a head number of 0 specifying the bottom surface, and 1 the top surface. For 640K L format discs, you can either specify the track and head as for the D format, or use a track number in the range 0 to 159. The sector number must be between 0 and 4 for D format discs, and 0 and 15 for the L format.

Once the sector editor is invoked, it is controlled in much the same way as the memory editor. The only new controls are the cursor keys together with Ctrl. These move backwards and forwards one sector (cursor left and right), or backwards and forwards a whole track (cursor up and down). These controls are summarised in the table. Bytes are edited in the same way, and the Tab key switches editing mode. The status line at the bottom displays the current head, track and sector, and also the disc address of the start of the current sector. Escape quits the editor. Before leaving a sector, whether moving onto another sector or quitting, the editor checks to see if any changes have been made. If they have, you will be asked whether to save the modified sector back to the disc. Pressing 'Y' will save it.

Key	-	+Shift	+Ctrl
←	back a byte	start of line	back a sector
→	forward a byte	end of line	forward a sector
↑	back 16 bytes	back a page	back a track
↓	forward 16 bytes	forward a page	forward a track

Listing 1.

```
240 MOV R0,#6:MOV R3,#&800
323 EQUS "Dedit":EQUB 0
324 ALIGN:EQUD dedit:EQUD &20001
325 EQUD desyn:EQUD dehlp
326 EQUS "DeditT":EQUB 0
327 ALIGN:EQUD deditTc:EQUD &40003
328 EQUD detsyn:EQUD dethlp
1397 .dehlp EQUS "*Dedit invokes the disc editor at a given disc address.":EQUB 13
1398 .desyn EQUS "Syntax: Dedit [<drive>] <disc address>":EQUB 0:ALIGN
1399 .dethlp EQUS "*DeditT invokes the disc editor at a given head/track/sector":EQUB 13
1400 .detsyn EQUS "Syntax: DeditT [<drive>] <head> <track> <sector>":EQUB 0:ALIGN
2654 EQUS "DiscShape":EQUB 0
```

RISC USER TOOLBOX (Part 3)



```

2655 EQUUS "DiscTrans":EQUB 0
2656 EQUUS "DiscUnTrans":EQUB 0
3712 B swi4:B swi5:B swi6
5895 MOV R6,#0:STR R6,[R12,#20]
6140 BNE C3:BL curdown:B keydone
6150 .c3 CMP R5,#&AF:BNE C4:BL curup
6160 B keydone:.c4 CMP R5,#&8C
6285 LDR R4,[R12,#20]:ORR R4,R4,#1
6286 STR R4,[R12,#20]
6551 LDR R4,[R12,#20]:ORR R1,R1,R4,LSL
#2
8260 .swi4 STMF D R13!,{R0,R3-R4,R14}
8270 CMP R0,#4:BCS swi4 hard:MOV R1,#1
8280 MOV R2,R0,LSL #29:MOV R4,#&400
8290 ADD R3,R12,#256:SWI "ADFS DiscOp"
8300 LDR R1,[R12,#768]:LDR R2,huge
8310 BIC R1,R1,#&FF:CMP R1,R2
8320 LDREQ R1,sh640:LDRNE R1,sh800
8330 LDREQ R2,si640:LDRNE R2,si800
8340 LDMFD R13!,{R0,R3-R4,PC}^
8350 .huge EQUD &67754800
8360 .si640 EQUD 640*1024
8370 .si800 EQUD 800*1024
8380 .sh640 EQUD &11008
8390 .sh800 EQUD &2050A
8400 .swi4 hard MOV R2,R0,LSL #29
8410 ORR R2,R2,#&C:MOV R4,#256
8420 ADD R3,R12,#256:MOV R1,#1
8430 SWI "ADFS_DiscOp":ADD R2,R12,#&2C0
8440 LDR R1,[R2,#16]:LDRB R0,[R2],#1
8450 LDRB R3,[R2],#1:ORR R0,R0,R3,LSL #
8
8460 LDRB R3,[R2],#1:ORR R0,R0,R3,LSL #
16
8470 MOV R2,R1:MOV R1,R0
8480 CMP R1,#0:ADREQ R0,nosherr
8490 LDMEQFD R13!,{R0,R3-R4,R14}
8500 ORREQS PC,R14,#1<<28
8510 LDMFD R13!,{R0,R3-R4,PC}^
8520 .nosherr EQUD &1003
8530 EQUUS "No shape map on hard disc"
8540 EQUB 0:ALIGN
8550 .swi5 STMF D R13!,{R1-R6,R14}
8560 MOV R0,R1,LSL #16:CMP R0,#2
8570 MOVEQ R4,R4,LSL #1:ADD R4,R4,R3
8580 MOV R3,R1,LSL #8:AND R3,R3,#&FF
8590 MLA R0,R4,R3,R5:AND R1,R1,#&FF
8600 MOV R0,R0,LSL R1:cmp R0,R2
8610 LDMLOFD R13!,{R1-R6,PC}^
8620 ADR R0,toobig:LDMFD R13!,{R1-R6,R1
4}
8630 ORRS PC,R14,#1<<28
8640 .toobig EQUD &1001
8650 EQUUS "Disc address too large"
8660 EQUB 0:ALIGN
8670 .swi6 STMF D R13!,{R4-R6,R14}

```

```

8680 CMP R0,R2:ADRHS R0,toobig
8690 LDMHSFD R13!,{R4-R6,R14}
8700 ORRHSS PC,R14,#1<<28
8710 AND R3,R1,#&FF:MOV R4,R0,LSR R3
8720 MOV R3,R4,LSL R3:SUB R3,R0,R3
8730 MOV R5,R1,LSR #8:AND R5,R5,#&FF
8740 MOV R6,#0:.swi6_2 SUBS R4,R4,R5
8750 ADDPL R6,R6,#1:BPL swi6_2
8760 ADD R2,R4,R5:MOV R5,R1,LSR #16
8770 MOV R1,#0:.swi6_3 SUBS R6,R6,R5
8780 ADDPL R1,R1,#1:BPL swi6_3
8790 ADD R0,R6,R5:LDMFD R13!,{R4-R6,PC}^
8800 .gpbuff STMF D R13!,{R0-R6,R14}
8810 AND R6,R1,#&FF:MOV R5,#1
8820 MOV R1,R3:ADD R3,R12,#1024:BIC R3,
R3,#15
8830 MOV R2,R4,LSL #29:ORR R2,R2,R0
8840 MOV R4,R5,LSL R6:SWI "ADFS_DiscOp"
8850 LDMFD R13!,{R0-R6,PC}^
8860 .gdrv STMF D R13!,{R2-R6,R14}
8870 CMP R1,R2:MOV R1,R0:BEQ gdrv2
8880 MOV R3,R1:SWI "ADFS_Drives"
8890 MOV R1,R3:LDMFD R13!,{R2-R6,PC}^
8900 .gdrv2 LDRB R0,[R1]:CMP R0,#ASC":"
8910 ADDEQ R1,R1,#1:LDRB R0,[R1]
8920 AND R0,R0,#&DF:CMP R0,#ASC"A"
8930 BCC gdrv3:CMPE R0,#ASC'I"
8940 BCS gdrv3:SUB R0,R0,#ASC"A"
8950 ADD R1,R1,#2:LDMFD R13!,{R2-R6,PC}^
8960 .gdrv3 MOV R0,#10:ORR R0,R0,#1<<29
8970 MOV R2,#7:SWI "OS_ReadUnsigned"
8980 ADD R1,R1,#1:MOV R0,R2
8990 LDMFD R13!,{R2-R6,PC}^
9000 .deditc STMF D R13!,{R14}
9010 LDR R12,[R12]:MOV R2,#2:BL gdrv
9020 MOV R3,R0:MOV R0,#16:SWI "OS_ReadU
nsigned"
9030 MOV R4,R2:MOV R0,R3:BL swi4
9040 STMF D R13!,{R0-R2}
9050 CMP R4,R2:ADRHS R0,toobig
9060 LDMHSFD R13!,{R14}
9070 ORRHSS PC,R14,#1<<28
9080 LDMFD R13!,{R0-R2}
9090 MOV R3,R4:B debit2
9100 .deditc STMF D R13!,{R14}
9110 LDR R12,[R12]:MOV R2,#4:BL gdrv
9120 MOV R3,R0:MOV R7,#3
9130 .deditc2 MOV R0,#10
9140 SWI "OS_ReadUnsigned":ADD R1,R1,#1
9150 STMF D R13!,{R2}:SUBS R7,R7,#1
9160 BNE deditc2:LDMFD R13!,{R4-R6}
9170 MOV R0,R3:BL swi4
9180 MOV R7,R1,LSR #16:cmp R6,R7
9190 BCS badpar:MOV R7,R1,LSR #8

```



RISC USER TOOLBOX (Part 3)

```

9200 AND R7,R7,#FF:CMPI R4,R7
9210 BCC deditc3:.badpar
9220 ADR R0,bperr:LDMFD R13!,{R14}
9230 ORRS PC,R14,#1<<28
9240 .bperr EQUD &1002
9250 EQUUS "Bad disc parameters"
9260 EQUB 0:ALIGN
9270 .dedittc3 MOV R3,R6
9280 STMF D R13!,{R0-R2}
9290 MOV R7,R5:MOV R5,R4:MOV R4,R7
9300 BL swi5:LDMVFS D R13!,{R1-R3,PC}:MO
V R3,R0
9310 LDMFD R13!,{R0-R2}:B dedit2
9320 .destat EQUUS " Drive 0 Head 0 Tr
ack 000 Sector 00 Address &0000000":EQ
UB 0
9330 .mkstat STMF D R13!,{R0-R4,R6-R8,R1
4}
9340 ADR R6,destat:ADD R7,R12,#256
9350 .mkstat2 LDRB R8,[R6],#1
9360 STRB R8,[R7],#1:CMPI R8,#0
9370 BNE mkstat2:ORR R4,R4,#ASC"0"
9380 ADD R7,R12,#256:STRB R4,[R7,#7]
9390 BL swi6:ORR R0,R0,#ASC"0"
9400 STRB R0,[R7,#15]:MOV R8,#24
9410 MOV R6,#&2F:.mkstat3 ADD R6,R6,#1
9420 SUBS R1,R1,#100:ADDMI R1,R1,#100
9430 BPL mkstat3:STRB R6,[R7,R8]
9440 ADD R8,R8,#1:MOV R6,#&2F
9450 .mkstat4 ADD R6,R6,#1:SUBS R1,R1,#
10
9460 ADDMI R1,R1,#10:BPL mkstat4
9470 STRB R6,[R7,R8]:ORR R1,R1,#ASC"0"
9480 ADD R8,R8,#1:STRB R1,[R7,R8]
9490 MOV R8,#36:MOV R6,#&2F
9500 .mkstat5 ADD R6,R6,#1:SUBS R2,R2,#
10
9510 BPL mkstat5:STRB R6,[R7,R8]
9520 ADD R2,R2,#58:ADD R8,R8,#1
9530 STRB R2,[R7,R8]:MOV R8,#55
9540 LDMFD R13,{R1}:mkstat6
9550 AND R0,R1,#15:MOV R1,R1,LSR #4
9560 CMP R0,#10:ADDCC R0,R0,#7
9570 ADD R0,R0,#48:STRB R0,[R7,R8]
9580 SUB R8,R8,#1:CMPI R8,#48:BNE mkstat
6
9590 MOV R5,R7:LDMFD R13!,{R0-R4,R6-R8,
PC}^
9600 .dedit2 MOV R4,R0:MOV R0,R3
9610 SWI &116:SWI &10C
9620 AND R5,R1,#&FF:MOV R6,#1
9630 MOV R6,R6,LSL R5:SUB R6,R6,#1
9640 AND R7,R0,R6:SUB R0,R0,R7
9650 .dedit3 STMF D R13!,{R0-R4}
9660 BL mkstat:MOV R3,#1:BL gpbuff
9670 ADD R0,R12,#1024:BIC R0,R0,#15:MOV

```

```

R2,R0
9680 RSB R4,R0,#0:AND R3,R1,#&FF
9690 ADD R0,R0,R7
9700 MOV R1,#1:MOV R1,R1,LSL R3
9710 ADD R3,R2,R1:MOV R1,#0
9720 .dedit4 STMF D R13!,{R2-R5}
9730 BL swi10:MOV R6,R2
9740 LDMFD R13!,{R2-R5}:CMPI R6,#27
9750 BEQ dedit6:CMPI R6,#&9C:BCC dedit5
9760 CMPI R6,#&A0:BCC dedit6
9770 .dedit5 AND R1,R1,#2:B dedit4
9780 .dedit6 AND R7,R1,#4
9790 LDMFD R13!,{R0-R4}
9800 CMPI R7,#0:BEQ dedit7
9810 BL dconf
9820 MOVNE R3,#2:BLNE gpbuff
9830 .dedit7 CMPI R6,#27:BNE dedit8
9840 SWI &11F:SWI &100:SWI &11F
9850 SWI &10A:LDMFD R13!,{PC}^
9860 .dedit8 MOV R5,#1:AND R7,R1,#&FF
9870 MOV R5,R5,LSL R7:MOV R7,R1,LSR #8
9880 AND R7,R7,#&FF:MUL R8,R5,R7
9890 AND R7,R0,#&E0000000
9900 BIC R0,R0,#&E0000000
9910 CMPI R6,#&9C:BNE tn1
9920 CMPI R0,R5:ADDCC R0,R0,R2:SUB R0,R0
,R5
9930 .tn1 CMPI R6,#&9D:ADDEQ R0,R0,R5
9940 CMPI R6,#&9E:ADDEQ R0,R0,R8
9950 CMPI R6,#&9F:BNE tn2
9960 CMPI R0,R8:ADDCC R0,R0,R2:SUB R0,R0
,R8
9970 .tn2 CMPI R0,R2:SUBCS R0,R0,R2
9980 BIC R0,R0,#&E0000000
9990 ORR R0,R0,R7:MOV R7,#0:B dedit3
10000 .dconf STMF D R13!,{R0-R2,R14}
10010 SWI &11F:SWI &100:SWI &11F
10020 SWI &111:SWI &183:SWI &111
10030 SWI &104:MOV R0,#75
10040 .dconf2 SWI &120:SUBS R0,R0,#1
10050 BNE dconf2:SWI &11F:SWI &100
10060 SWI &11F:SWI "OS_WriteS"
10070 EQUUS "Sector modified - Rewrite? "
10080 EQUB 7:EQUB 0
10090 MOV R0,#15:MOV R1,#1:SWI "OS_BytE"
10100 .dconf3 SWI "OS_ReadC":BCS dconf4:
AND R0,R0,#&DF
10110 CMPI R0,#ASC"N":CMPI R0,#ASC"Y"
10120 BNE dconf3:SWI "OS_WriteC":B dconf5
10130 .dconf4 MOV R0,#&7E:SWI "OS_BytE"
10140 MOV R0,#ASC"N":SWI "OS_WriteC"
10150 .dconf5 SWI &111:SWI &180:SWI &111
:SWI &107
10160 CMPI R0,#ASC"N":LDMFD R13!,{R0-R2,P
C}

```

Too early to think of Christmas shopping?

Not when it
comes to
our show!

New Horticultural Hall
Greycoat Street
London SW1

10am-6pm Fri, Nov 11
10am-6pm Sat, Nov 12
10am-4pm Sun, Nov 13

This show has it all!

LOADSA exhibitors (around 70)
LOADSA hardware
LOADSA software
LOADSA new products
LOADSA games
LOADSA happenings
LOADSA technical advice

... and most important of all for you, the visitor – **LOADSA BARGAINS!**

With hundreds of special show offers available you could end up a financial winner.

You can even save £1 a head before you get there by using this advanced ticket form.



The 1988 Innovation Awards

- Take a stroll down Innovation Row – a brand new show feature area, specially constructed for the event.
- See the grand finalists displaying their innovations in public for the first time.
- Help Pick The Winners. You will be able to cast a vote in both categories of the awards – the BBC Micro and the Archimedes.

No matter which Acorn machine you use – from the Electron up to the top of the range Archimedes – you'll find just what you are looking for at the show.

All the leading companies servicing each sector of the Acorn market will be out in force to demonstrate their latest developments.

Traditionally the liveliest Acorn event of the year, this pre-Christmas show is the one you can't afford to miss.

Advance ticket order

SAVE £1

ELECTRON & BBC MICRO USER SHOW

Please supply:

Adult tickets at £4 (save £1) £
 Under-16s tickets at £2.50 (save £1) £
 Cheque enclosed made payable to Database Exhibitions Total £ _____

Please debit my credit card account: Access Visa Expiry date: / /

Name
Address

Signed

Post to: Database Exhibitions, Europa House, Adlington Park, Adlington, Macclesfield SK10 4NP.

PHONE ORDERS: Ring Show Hotline: 0625 879920
PRESTEL ORDERS: KEY *89, THEN 614568383
MICROLINK ORDERS: MAILBOX 72: MAG001
Please quote credit card number and full address

DATA BASE EXHIBITIONS

How to get there

Underground: The nearest tube stations are VICTORIA (Victoria, District & Circle Lines). St JAMES'S PARK (District & Circle Lines) and PIMLICO (Victoria Line).

By British Rail: VICTORIA STATION. The halls are a 10 minute walk from the station.

By Bus: 11, 24, 29, 70, 76 and Red Arrow 507 to Victoria Street – alight Army and Navy Stores.

FAST MODULE UTILITY

David Spencer offers a short utility to increase the speed of your Archimedes.

As most readers will be aware, nearly all of the major functions of the Arthur operating system are carried out by *Relocatable Modules*. These modules are simply programs in ARM machine code which are written in such a way that they can integrate with the operating system. There are twenty three modules contained within the operating system ROMs, and these include for example Basic, and the ADFS. In addition to the standard modules, it is also possible to load modules into RAM, and have them treated in exactly the same way as the ROM based ones. Furthermore, it is possible to load into RAM a module which already exists in ROM, automatically disabling the old version. A major advantage of having modules in RAM is that because of the way the Archimedes accesses memory, they will run about 22% faster than their ROM based equivalents.

It is because of the speed difference between RAM and ROM modules, that RAMBASIC is supplied on the Archimedes Welcome disc. This is a module containing a version of Basic which is identical to that in the operating system ROM. However, the module loads into RAM, disabling its ROM based counterpart, and therefore runs quicker.

Obviously, it would be possible to have disc copies of all the operating system modules, and when you want a particular module to run faster, just load it from disc. This does however seem a very clumsy approach to adopt when the modules are already sitting there in the computer. The new Arthur 2 operating system provides a star command, *RMFASTER, which takes a module from ROM and copies it into RAM. The ROM version is then disabled, and the new RAM copy started up in its place. The program given here generates a disc based star command which performs exactly the same function for Arthur 1.20.

The listing below should be typed in and saved. When the program is run, it will create the *RMFASTER command, and save it to disc.

```
10 REM >FastSrc
20 DIM code 100
30 FOR pass=0 TO 3 STEP 3
40 P%:code
50 [OPT pass
60 MOV R0,#18:SWI "XOS_Module":MOVVS
PC,R14
```

```
70 MOV R1,R3:SUB R0,R1,#4
80 LDR R2,[R0]
90 MOV R0,#11:SWI "XOS_Module"
100 MOV PC,R14
110 ]NEXT
120 SYS "OS_File",10,"RMFaster",&FFC,,code,P%
```

The new command has the syntax:

*RMFASTER <module>

where <module> is the title of one of the ROM modules. These can be found out by typing *MODULES. For example:

*RMFASTER BASIC

will make a fast RAM copy of Basic.

Because the command copies a module into RAM, it is possible to get the error 'No room in RMA', if there is not enough memory in which to store the new copy. The way around this is to type QUIT first, in order to return to Arthur's ** prompt, before issuing the *RMFASTER command. This is also necessary if you are making a copy of the current application, e.g. Basic, because these cannot be copied while they are actually running.

HOW IT WORKS

The operation of the *RMFASTER command relies on the operating systems SWI "OS_Module", which is used to perform various operations on relocatable modules.

Line 60 of the program uses this call to find out information about a particular module. When the command is run, the operating system passes a pointer to the rest of the command line in register R1. This is passed to SWI "OS_Module" as a pointer to the module name. The call returns, among other things, the memory address of the start of the module in R3.

Lines 70 and 80 put the start address of the module in R1, and the length of the module in R2. The length of every module is stored in the word before the start of the module, and can therefore be read with a simple LDR instruction.

Line 90 uses a different form of SWI "OS_Module", which this time copies the area of memory starting at the address in R1 and of length R2 into the relocatable module memory, and then starts up the new module.



Communications software reviewed by Ray Hughes.

After the success of the public domain version of ARCterm for communications work on the Archimedes, we are finally blessed with a full-blown commercial version from the same author. The package consists of a disc accompanied by a manual.

ARCterm is a large program, but can best be visualised in several distinct sections.

The Terminal

This is the main part of the ARCterm software suite, and enables you to call the other program sections as and when required.

The Mailbox/Frame Editor

This section allows you to create messages off-line that you may later wish to send to Prestel or a similar viewdata system.

The Script Interpreter

Scripts are programs written in a language called Arcscript, and can be used to control almost all the functions that the program is capable of. I am sure that the provision of this facility is going to lead to the long term success of this software. Many people will be creating their own script files and either selling them or putting them into the public domain on bulletin boards etc.

The Host Mode

With this, simple Bulletin boards or message systems may be set up by the user on his own machine (provided he has a suitable modem). Messages can be left or retrieved, and files can be uploaded or downloaded, all without any user interaction.

THE SOFTWARE

The program itself is written in a mixture of ANSI C and ARM machine code (for any time-critical routines). The pre-release version of the manual that I had was well written, and even contained a useful glossary of the more commonly used jargon words. The manual covers all that an experienced user would need

to make full use of the functions, but it should be appreciated that users fairly new to comms are going to find it all quite heavy going at first.

The author has an oft protested dislike of desktop type operating systems, and hence all commands in ARCterm are entered from the keyboard, preceded by the use of the Alt key. However, one can use one of the Alt-Toggle commands (Alt TM) to enable the mouse to be used on-screen for selecting frame numbers etc.

FILE TRANSFER PROTOCOLS

One of the main uses for any communications software is to allow the transfer of files over the public telephone network. To avoid corruption of the transmitted files, a method of checking each transfer needs to be employed. This can be one of the most complex areas as far as computer to computer communications is concerned, but all that concerns us here is that ARCterm has the means to use most of the available file-transfer protocols in common use today. Those available are:

XMODEM	XMODEM CRC
CET tsw	KERMIT
ZMODEM	MBXfer
ARCfer	

Most of the above will be well known to any regular comms users, but MBXfer and ARCfer are specific to ARCterm and need a brief description.

MBXfer

This allows the sending, via a Prestel mailbox, of programs and other such files. However, such is the usual slowness of the Prestel system that it would really only be suitable to send small files by this method.

ARCfer

ARCterm users may send files to each other while 'chatting' via the keyboard. Thus users may 'talk' and transfer data at the same time.

THE ARCSRIPT LANGUAGE

Having an inbuilt programming language is not a new idea for communications software. The language currently contains 44 keywords. Some are useful for general programming, while others are very specific to a communications environment.

The addition of an inbuilt language of this type does mean that the program can be totally automated to perform the most complex communications tasks at the simple press of a couple of keys. For example, I quickly typed in a program using TWIN (the final version of ARCterm will contain its own editor) to enable me to log on to the Prestel database and automatically send my ID and password.

ARCTERM IN USE

Since receiving my pre-release version of the software I have used it many times for logging onto Prestel, Telecom Gold and various bulletin board services, and apart from a few known failings the software performed well in all cases.

The Prestel emulation continues the excellent trend started with the public domain version of ARCterm of avoiding the use of the Archimedes mode 7 character set, which leaves much to be desired.

So how does one actually go about getting onto Prestel for instance. Well firstly just IBOOT the supplied disc to load the main terminal program. The screen display then changes to show all relevant information on the top 2 lines of the screen, with the rest of the screen area being reserved for data reception.

The information on these two top lines is quite comprehensive and contains the following data:

Current terminal emulation

Transmit baud rate

Receive Baud rate

Word format

NL=CRLF : sequence required

Local echo status

Remote echo status

Current time

Filter setting
Spool status
Printer output status

And additionally in the final version:
Phone number connected to
On-line time (continually updated)

FUTURE ENHANCEMENTS

The description I have given so far is, as I have said, based on the pre-release version of the software. But by the time you read this it may well be in its final release form. After some extensive on-line "chats" with the author I feel assured that the final version really will do all that could be asked of a comms package. However, a number of features were incomplete and could not be tested at the time of this review. These are detailed below.

Script Learn

Auto generation of script files by saving your keystrokes.

Mailbox Editor and Uploader

Reduce on-line charges by creating your messages off-line.

Notepad

A pop-up notepad to allow the quick jotting down of phone numbers etc.

Alarm

To remind you to save some money on the bill (among other applications)!

ARCfer & MBXfer

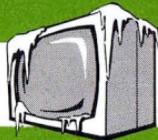
These items, mentioned earlier, had not been finished at the time of writing.

IN CONCLUSION

ARCterm is one of the first comms packages to be released for the Archimedes, and the one against which I am sure other such software will be judged.

RU

Product	ARCterm Professional
Supplier	The Serial Port
	12 Housman Road,
Price	Street, Somerset BA16 OSD. £49.49 inc VAT.



SCREEN FREEZER AND DUMPER (Update)

David Spencer offers an improvement to this handy module published in last month's RISC User.

As it stands, the Screen Freezer and Dumper module stops a program when the two ALT keys are pressed together, and then waits for another key to be pressed. If this key is 'S', then the screen is saved, otherwise execution continues. However, on the rare occasion when there are already any key presses in the keyboard buffer when the program is frozen, confusion can arise. In this case, the next keypress will be read out of the buffer, rather than waiting for a key to be pressed. We should stress here that the author's original version did not suffer from this weakness.

The modifications presented here change the module so that it specifically waits for a key to be pressed, rather than taking any keys out of the buffer. The lines to change are:

```
850 SWI "OS_IntOn"
851 .kloop MOV R0, #&7A:SWI "OS_Byt"
```

```
852 CMP R1, #&FF:BEQ kloop
853 MOV R0, #&78:ORR R1, R1, #&80:SWI
      "OS_Byt"
860 CMP R1, #209:BNE nosave
```

Once these lines have been entered, the program should be saved and then the module assembled and used as described in the original article. This month's RISC User disc contains the new version of the Freezer program.

The new version of the module scans the keyboard until a key is pressed, and then sets that key as the last key pressed. This prevents the operating system from subsequently entering the key into the keyboard buffer. Finally, the value of the key pressed is checked to see whether a screen save is required.

RU

3D LANDSCAPE EDITOR (continued from page 6)

```
800 PRINTTAB(G%*3)D%(G%,21-F%) " ";
810 NEXT:NEXT
820 ENDPROC
830 :
840 DEF PROCdisplay
850 CLS:new%==FALSE:ORIGIN 0,64
860 FOR G%=19 TO 1 STEP -1
870 FOR F%=1 TO 19
880 CASE SGN(Y%(F%,G%)-Y%(F%+1,G%)) OF
890 WHEN 1:GCOL colour(RND(11)+6) TINT
(RND(4)-1)*64
900 WHEN 0:GCOL colour(RND(7)+3) TINT
(RND(4)-1)*64
910 WHEN -1:GCOL colour(RND(6)) TINT (R
ND(4)-1)*64
920 ENDCASE
930 PLOT &54,X%(F%,G%),Y%(F%,G%)
940 PLOT &54,X%(F%+1,G%),Y%(F%+1,G%)
950 PLOT &55,X%(F%,G%+1),Y%(F%,G%+1)
960 PLOT &55,X%(F%+1,G%+1),Y%(F%+1,G%+
1)
970 GCOL 0:MOVE X%(F%+1,G%+1),Y%(F%+1,
G%+1)+4:DRAW X%(F%+1,G%),Y%(F%+1,G%)+4:D
RAW X%(F%,G%),Y%(F%,G%)+4
980 NEXT:NEXT:ORIGIN 0,0
990 ENDPROC
1000 :
1010 DEF PROCedit
1020 *FX21,9
1030 *POINTER
1040 MOUSE RECTANGLE 48,358,954,630
1050 PROCprint values
1060 REPEAT:exit%==FALSE
1070 MOUSE x%,y%,b%
1080 x%==x%/48:y%=(1023-y%)/32
1090 CASE b% OF
1100 WHEN 4:PROCchange(1)
1110 WHEN 2:PROCchange(-1)
1120 WHEN 1:exit%==TRUE
1130 ENDCASE
1140 TIME=0:REPEAT UNTIL TIME>8
1150 UNTIL exit%:MOUSE OFF
1160 ENDPROC
1170 :
1180 DEF PROCchange(p%)
1190 q%==D%(x%,21-y%):q%+=p%
1200 IF q%<0 q%=0 ELSE IF q%>9 q%==9
1210 D%(x%,21-y%)=q%
1220 PRINTTAB(x%*3,y%)q%
1230 new%==TRUE
1240 ENDPROC
```

RU

ADVERTISING IN RISC USER

RISC User has the largest circulation of any magazine devoted to the Archimedes. In fact, we believe over 80% of Archimedes owners are regular readers.

RISC User is therefore the ideal medium for advertising all products for the Archimedes to a discerning and committed readership. With nine issues of RISC User successfully complete we can now offer advertising space at attractive rates.

To find out more, contact

Yolanda Tureuo
on (0727) 40303
or write to
RISC User, Dolphin Place,
Holywell Hill, St Albans,
Herts AL1 1EX.

ARCHIMEDES RAM/ROM PODULE AND BACKPLANE

Before you can fit any type of podule to an Archimedes (305 or 310), a backplane must be installed. Our fully guaranteed backplane is a high quality unit offering the following features:

- ★ Easy to fit - no soldering
- ★ Accepts two podules
- ★ Approved by Computer Concepts for use with their RAM/ROM podule
- ★ Only £41.75

Backplane + Computer Concepts podule £94.00
As above with battery backup £103.00
Standard Computer Concepts podule £53.00
As above with battery backup £63.50

Send cheques, POs or official orders to the address opposite. Prices include VAT. N.B. Quote your RISC User membership number when ordering to obtain a discount of £2.50 on the podule and backplane combination!



IFEL
36 Upland Drive
Derriford
Plymouth PL6 6BD
(07555) 7286

ARCHEFFECT

FX

VISUAL EFFECTS
MODULE

ARCHEFFECT is an easy to use image manipulation package. By the use of simple *Commands you will be able to quickly produce spectacular visual effects, similar to those seen on TV. It also allows for the easy use of fonts.

The module has been specifically designed for use in the high resolution graphics modes to take full advantage of the potential of the Archimedes.

In addition to the many image manipulation commands the module provides an alternative method of using the powerful font manager provided on the Archimedes, bypassing the complex 'SYS' commands previously necessary.

ARCHEFFECT - £24.95 inc VAT and p&p

Please send me copies of Archeffect at £24.95 each.

I enclose a cheque or postal order for the sum of £.....

Please make cheques or postal orders payable to FX.
FX, 207 South Avenue, Southend-on-Sea, Essex SS2 4HT.
(For further details send SAE)



Postbag

We welcome your letters for publication on our Postbag page whether it be comments on the magazine and the Archimedes, technical queries or information for other readers.

PC EMULATOR AND ARTHUR

Now that Acorn has made the PC Emulator 40% faster, is it now as fast as a standard PC/XT? Are Acorn still planning to produce a PC podule? Also, is Arthur 1.2 the final operating system, or will Acorn be releasing a new one? What happened to the much rumoured about ARX?

Karl Strickland

The speed of the PC emulator depends a lot on what tasks are being performed. Functions such as disc access, which can take advantage of the ARM's speed, will be faster on the emulator than with a real PC. Other operations will be slower. However, overall, the speed of the emulator will be much the same as that of a standard PC. Acorn have dropped plans to produce a hardware PC podule, because the design they were working on has been superseded by cheap AT clones. It remains to be seen whether a third party steps in to produce a similar podule.

On the subject of operating systems, Acorn are developing an improved version of Arthur, called Arthur 2. This is however not fully compatible with the current version, and will almost certainly be sold as a separate product. It is believed that Acorn has scrapped ARX, a UNIX look-alike, and is instead concentrating on the genuine article for its ARM based successor to the Archimedes.

PRINTER DRIVERS

I have recently been using Acorn's 1st Word Plus quite successfully for a variety of tasks. However, I needed some way of printing the output on the Xerox X3700 laser printer at the computer centre where I work.

Constructing a printer driver for the laser printer was not really a problem; the manual was easy to follow, and most of the features of 1st Word Plus were straightforward to implement. With a bit of fiddling about 90% of the Archimedes Latin1 set has been covered. Although proportional text is not handled, fixed fonts suffice. The main problem was to persuade Arthur to send printer output to a named file. If this could be achieved then it

would be a simple matter to transfer this file to our VAX computer and print it on our X3700.

The clue came in the *Programmer's Reference Manual* where the system variable PrinterType\$<n> is mentioned, and is used to connect VDU print streams to files. If therefore PrinterType\$4 is *SET to :0.spooler, say, then any printing destined for a network is sent to :0.spooler. It is therefore a simple matter in 1st Word Plus to switch to the network option before printing, select the X3700 printer driver, and have one's masterpiece ready for the laser printer. Although I haven't tried it, I assume that the same technique would work with most other word processors as well.

Gary Pike

CROSSFADE MODS

The CrossFade program (RISC User Issue 7) failed to run on my Archimedes 440. Investigation showed that it relied on absolute screen addressing. Can you insist that your authors don't cut corners - it's so easy to do it all properly on the Archimedes. Anyway, listed below are the changes to make the program work on any Archimedes with Screensize set to give 160Kbytes of screen space (SCREENSIZE 20 on 300 series, 5 on the 400 series).

Delete lines 580, 590 and 600 and replace with:

```
582 DIM input% 7, output% 3
584 input%!0=149
586 input%!4=-1
588 SYS "OS_ReadVduVariables", input%
, output%
590 screen_address1!=output%
592 screen_address2!=output%+80*1024
594 OSCLI("LOAD SCREEN2"+STR$-
(screen_address2))
596 A%<screen_address2: C%<80*64
598 B%<screen_address1: E%<63
```

The important point is that using OS_ReadVduVariables allows the start of screen memory to be determined by the program.

Sean Kelly

Thank you to Sean for this modification. While we always endeavour to make programs as 'legal' as possible, this is not always practical, especially when it would greatly increase the length of a program.

RU

More useful hints and tips rounded up by David Spencer.

TWIN HINTS

Lee Calcraft

Here are a couple of useful search and replace strings to use with Twin. The first is:

Most space Exact \ Many Any Newline By Newline

(where 'space' means one press of the space bar), which removes comments from an assembler program. To remove any comments starting with a semi-colon (rather than a '\'), replace the '\' with a '. The second sequence concatenates lines of Basic or assembler by replacing a carriage return, and any surrounding spaces, by a colon:

Many space Newline Many space By Exact :
This sequence works with the TWINO 8 option only.

DISASSEMBLER FIX

David Spencer

The debugger module included in Arthur features a SWI call to disassemble an ARM instruction. This call has the name:

`'Debugger_Disassemble'`

and is SWI number &40380. The call is made with the instruction to be disassembled in register R0, and returns with R1 pointing to the disassembled string, and R2 containing its length. However, there is a bug which crops up when disassembling branch instructions. As all branches are relative to the program counter, you can only disassemble a branch properly if you know the address that the instruction came from. The Debugger doesn't have this information, and instead disassembles all branches relative to the start of the Debugger module.

The Basic function given here fixes this shortcoming. The function is called with an address as the parameter, and then takes the instruction from this address and disassembles it. The result returned is the disassembled instruction as a string. All branches are disassembled relative to the address from which the instruction is fetched. The function works by using the Debugger's SWI call, and then checking the result to see if it is a branch. If so, the address in the result string is modified. The address at the end of line 1070 is the start address of the Debugger module, and is correct for Arthur 1.20.

```
1000 DEF FNdisass(P%)  
1010 LOCAL A%,B%,A$  
1020 SYS "Debugger_Disassemble",!P%  
      TO ,A%,B%
```

```
1030 A$="":WHILE B%  
1040 A$=A$+CHR$?A%:A%+=1:B%-=1  
1050 ENDWHILE  
1060 IF LEFT$(A$,1)="B" AND MID$(A$,2,1)  
    <"I" THEN  
1070 A%=$EVAL MID$(A$,9)-&38486A0  
1080 A$=LEFT$(A$,9)  
1090 A$=A$+RIGHT$("0000000"+STR$~(A%+P%  
    ,8)  
1100 ENDIF  
1110 =A$
```

TYPING FILES

N. Kirby

All files on the Archimedes can have a filetype associated with them. This is a three digit hexadecimal number that gives the operating system, and the user, some idea of the information that the file might contain. By using filetypes, it is possible to print to the screen the contents of a text file simply by typing:

`*<filename>`

Filetypes are in the range 0 to &FFF. Those from 0 to &7FF are allocated for the user. For example we could assign a file to filetype &7FF using the command:

`*SETTYPE <filename> 7FF`

You then need to tell Arthur how to handle such files, by issuing the command:

`*SET Alias$@RunType_7FF TYPE %`

which tells the operating system to print the contents of any file with filetype &7FF that is *RUN.

BOOTING THE DISC AROUND

Andrew Benham

There seems to be a great deal of confusion over auto-booting discs on the Archimedes. On previous BBC computers, using Shift-Break to boot a disc would run the file \$.!BOOT on either the current drive for the ADFS, or drive 0 for the DFS. However, on the Archimedes Shift-Break accesses the file &.!BOOT: that is the !BOOT file in the User Root Directory (URD). As the URD can be changed with *URD <directory>, it is possible to have many boot files on each disc, and because the URD does not have to be on the current drive, you can have the system boot from any drive. The only problem is that the *OPT4 setting applies to *all* the !BOOT files on a particular disc, regardless of their type.

DISC MENU FILETYPES

N. Kirkby

The RISC User Disc Menu (Issues 1 & 2) only takes account of the bottom 2 digits of a filetype, and the top digit must be &F. Therefore, any new filetype must be in the range &F00 to &FF5. To use type &FF5, for example, with a TEXT file, you should include the following line in the menu program.

```
5845 EQUB &F5:EQUS " TEXT"
```

(There are six spaces before the word TEXT.) If you use a different filetype, change the value of the EQUB.

DEBUGGING BASIC

Dennis Weaver

The debugging of Basic programs can be made a much less painful task by careful use of a couple of Basic commands. The first of these is LVAR, which doesn't take any parameters. Typing LVAR will produce a list of all the variables defined, along with their values, except for arrays, in which case their sizes are given. But, more importantly, LVAR also lists the names of all functions

and procedures that have been called by the program, and displays the first line of any libraries installed. By using LVAR when a program fails, it is possible to check the state of the variables, and functions and procedures, against their expected values. The other useful command is LIST IFDEF, which will list all lines in a program that contain a DEF. This means that you can use it to find the definitions of all the procedures and functions in a program, regardless of whether they have been called or not. Similarly, LISTIFPROC will find both procedure definitions, and all the lines on which the procedures are called.

VIEWSTORE INDEXES

Ian Stubbs

If you are running Viewstore under the 6502 emulator, it is possible that you will get an 'Address exception' error when trying to use the index facility. The simple solution is to increase the Systemsize configuration to at least 1, to provide more workspace for the emulator.

RU

PLOTTING WITH MINERVA (continued from page 29)

MACROS

One feature especially worthy of mention is the macro facility. A file of commands can be created to load data, and create graphs, and again a complete sequence of graphs can be controlled in this way. This is very useful for a presentation of any kind using the capabilities of GammaPlot in a more dynamic way.

OTHER FACILITIES

A statistics option provides comprehensive (and fast) analysis of any data, including mean, standard deviation and correlation. The print options (from several menus) support Epson FX, Epson JX (colour), Integrex 132 (colour) and compatible printers.

CONCLUSIONS

I am pleased to report that I feel quite enthusiastic about GammaPlot. But there are limitations. It does not appear possible to display multiple line charts, segmented bar charts or clustered histograms, i.e. to display, for comparison, several data sets on the same graph or chart. Similarly, there is no easy way

to scale accurately comparable data (for example share prices of different companies), to compare trends. So don't take my enthusiasm to mean that this package will do everything.

The manual is better than previous ones from Minerva, but I still feel that the printing is just too dense, and the appearance is still daunting, a pity for such a delightfully easy-to-use package.

If GammaPlot satisfies your needs, then what it does it does well. The various menu screens are excellent examples of their kind and the whole system is easy to learn and easy to use. What more could you want?

Product	GammaPlot
Supplier	Minerva Software
	69 Sidwell Street, Exeter, Devon EX4 6PH.
Price	Tel. (0392) 437756 £69.95 inc. VAT

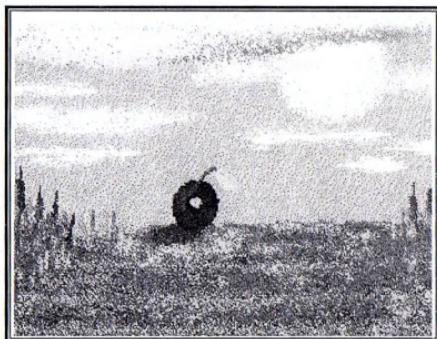
RISC User Magazine Disc

September 1988

CONTENTS

LANDSCAPE EDITOR - Design your own Zarch look-alike colourful 3D landscapes with this simple program. As a special bonus, four sample landscapes are included.

MUSIC MAESTRO - The perfect complement to the Music Editor on the Welcome Disc, allowing tunes to be played without having to load the Editor.



ARCHIMEDES VISUALS

Another two graphics orientated programs. The first is a complete art package with air brush in just twenty lines. A sample of what can be produced is also featured. The second is a program to draw cones and cylinders using a dithering effect.

FAST IMAGE RESCALER

This incredibly fast routine from the author of the Superfast Mandelbrot Generator (RISC User Disc Issue 8), lets you rescale and move any area of the screen in real-time. A full demonstration is also included.

SCREEN FREEZER UPDATE

An enhanced version of last month's Screen Freezer module.

ARCHIMEDES ANIMATION

Continuing on the topic of animated spheres, the three programs this month show how to produce the effect of a moving ball.

READING FROM THE ADFS

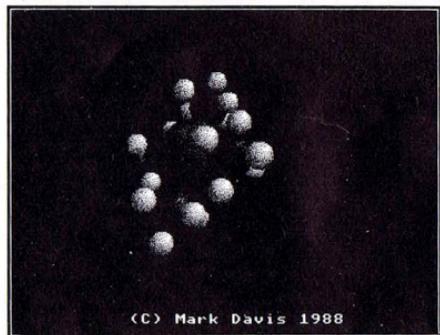
Continuing from last month, a complete file-finder for the ADFS or Econet.

ARM ASSEMBLER

A screen mover to demonstrate the theory covered in this month's article.

RISC USER TOOLBOX

This month's addition to the Toolbox is a complete disc sector editor.



★★★ And as a special bonus ★★★

SNOOKER TABLE

A short program which uses the Cones and Cylinders routine from 'Archimedes Visuals' to draw a snooker table.

ROTATING SPHERE DEMONSTRATION

A machine code program which draws orbiting spheres in real time.

MOZART ALLEGRO

A Music Editor tune which can be played with the Music Maestro.

All RISC User magazine discs are available to order, or by subscription. Full details of prices etc are given on the back cover of each issue of RISC User.

RISC USER magazine

MEMBERSHIP

RISC User is available only on subscription at the rates shown below. Full subscribers to RISC User may also take out a reduced rate subscription to BEEBUG (the magazine for the BBC micro and Master series).

All subscriptions, including overseas, should be in pounds sterling. We will also accept payment by Connect, Access and Visa, and official UK orders are welcome.

RISC USER SUBSCRIPTION RATES

£14.50	1 year (10 issues) UK, BFPO, Ch.I
£20.00	Rest of Europe & Eire
£25.00	Middle East
£27.00	Americas & Africa
£29.00	Elsewhere

RISC USER & BEEBUG

£23.00
£33.00
£40.00
£44.00
£48.00

BACK ISSUES

We intend to maintain stocks of back issues. New subscribers can therefore obtain earlier copies to provide a complete set from Vol.1 Issue 1. Back issues cost £1.20 each. You should also include postage as shown:

Destination	UK, BFPO, Ch.I	Europe plus Eire	Elsewhere
First Issue	40p	75p	£2
Each subsequent Issue	20p	45p	85p

MAGAZINE DISC

The programs from each issue of RISC User are available on a monthly 3.5" disc. This will be available to order, or you may take out a subscription to ensure that the disc arrives at the same time as the magazine. The first issue (with six programs and animated graphics demo) is at the special low price of £3.75. The disc for each issue contains all the programs from the magazine, together with a number of additional items by way of demonstration, all at the standard rate of £4.75.

MAGAZINE DISC PRICES

	UK	Overseas
Single issue discs	£ 4.75	£ 4.75
Six months subscription	£25.50	£30.00
Twelve months subscription	£50.00	£56.00

Disc subscriptions include postage, but you should add 50p per disc for individual orders.

All orders, subscriptions and other correspondence should be addressed to:

RISC User, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX.

Telephone: St Albans (0727) 40303

(24hrs answerphone service for payment by Connect, Access or Visa card)